PLANNING AND CONTROL OF ROBOTIC MANIPULATION ACTIONS FOR EXTREME ENVIRONMENTS

by

TOMMASO PARDI

A thesis submitted to
The University of Birmingham
for the degree of
DOCTOR OF PHILOSOPHY

School of Metallurgy and Materials
College of Engineering and Physical Sciences
The University of Birmingham
8th September 2022

Abstract

A large societal and economic need arises for advanced robotic capabilities, where we need to perform complex human-like tasks such as tool-use, in environments that are hazardous for human workers. This thesis addresses a collection of problems, which arise when robotic manipulators must perform complex tasks in cluttered and constrained environments. The work is illustrated by example scenarios of robotic tool use, grasping and manipulating, motivated by the challenges of dismantling operations in the extreme environments of nuclear decommissioning

Contrary to popular assumptions, legacy nuclear facilities (which can date back three-quarters of a century in the UK) can be highly unstructured and uncertain environments, with insufficient *a-priori* information available for *e.g.* conventional pre-programming of robot tasks. Meanwhile, situational awareness and direct teleoperation can be extremely difficult for human operators working in a safe zone that is physically remote from the robot. This engenders a need for significant autonomous capabilities. Robots must use vision and sensory systems to perceive their environment, plan and execute complex actions on complex objects in cluttered and constrained environments. Significant radiation, of different types and intensities, provides further challenges in terms of sensor noise. Perception uncertainty can also result from *e.g.* vision systems observing shiny featureless

metal structures. Robotic actions therefore need to be: i) planned in ways that are robust to uncertainties; and ii) controlled in ways which enable the robust reaction to disturbances.

In particular, we investigate motion planning and control in tasks where the robot must: maintain contact while moving over arbitrarily shaped surfaces with end-effector tools; exert forces and withstand perturbations during forceful contact actions; while also avoiding collisions with obstacles; avoiding singularity configurations; and increasing robustness by maximising manipulability during task execution. Furthermore, we consider the issues of robust planning and control with respect to uncertain information, derived from noisy sensors in challenging environments.

We explore the Riemannian geometry and robot's manipulability to yield path planners that produce paths for both fixed-based and floating-based robots, whose tools always stay in contact with the object's surface. Our planners overcome disturbances in the perception and account for robot/environment interactions that may demand unexpected forces. The task execution is entrusted to a hybrid force/motion controller whose motion space behaves with compliance to accommodate unexpected stiffness changes throughout the contact.

We examine the problem of grasping a tool for performing a task. Firstly, we introduce a method for selecting the grasp candidate onto an object yielding collision-free motion for the robot in the post-grasp movements. Furthermore, we study the case of a dual-arm robot performing full-force tasks on an object and slippage on the grasping is allowed. We account for the slippage throughout the task execution using a novel controller based on the sliding mode controllers.

Acknowledgements

The production of this thesis wouldn't have been possible without the support of my supervisors and my wife.

I also want to thank the Nuclear Decommissioning Authority for funding the work presented in this thesis and the National Nuclear Lab Ltd for providing industrial support and discussions. The insights from these industrial partners enabled this work to progress toward a direction that might serve industrial applications and academic research.

Contents

List of Figures

CHAPTER 1

Introduction

1.1 Towards the robotic automation in extreme environments

The disposal of radioactive wastes from the decommissioning of nuclear power plants is one of the biggest world-class challenges in the last decades. Today, over 60 countries produce energy from nuclear sources, and more than 400 reactors are currently active. After an estimated life-cycle of 60-70 years, every power plant must undergo decommissioning to reuse the site and remediate the environmental damages caused by the radiations, [Ass20]. Additional 182 reactors are already permanently shut down around the world and are at several decommissioning status.

In the UK, a total of 17 nuclear sites are undertaking the decommissioning process, and the Nuclear Decommissioning Authority (NDA) estimates the cleanup will take around 120 years, with a costly endeavour of over £130 billion for UK's citizens, [Aut19].

Of the forecasted amount, a substantial component accounts for the cleaning up of Sellafield. This site has facilities dating back to the 1940s and covers a region of around

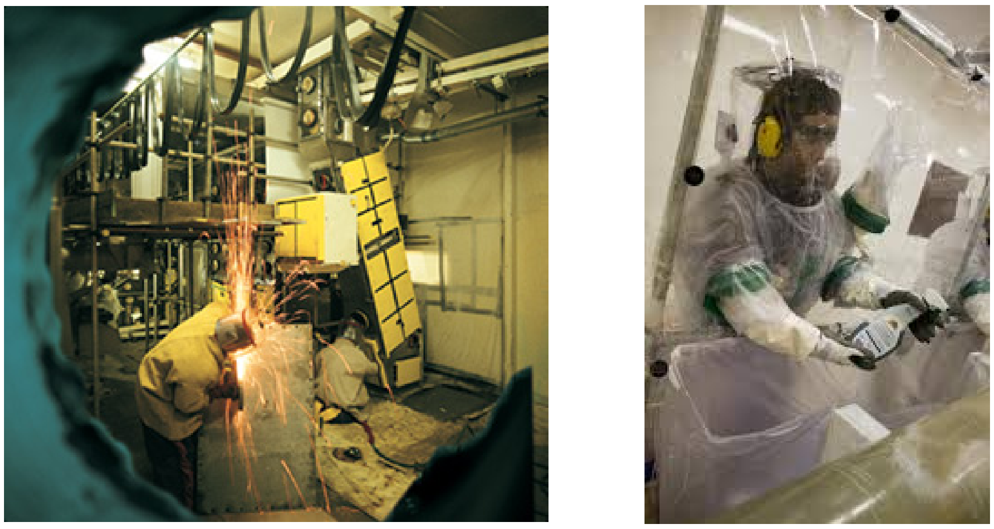

(a) (b)

Figure 1.1: Nuclear decommissioning worker wearing an air-fed plastic suit underneath a heavy leather overcoat, and multiple layers of gloves, while using power tools to cut legacy nuclear plant contaminated by alpha-radiation emitting substances, such as plutonium dust. The leather coat protects the plastic suit from being punctured by hot sparks during cutting. Maximum 2hrs work per day is possible, due to extreme discomfort and heat exhaustion as the suit fogs and fills with sweat. Image courtesy of Sellafield Ltd.

$6km^2$ housing thousands of buildings. Throughout the years, the compound served several purposes, which have ranged from the creation of material for weapons to nuclear fuel reprocessing.

In the early years, the cold war hasted the construction of these buildings, which resulted in shortsighted planning of the site's life-cycle. Nowadays, most blueprints are missing, and we lack a record of waste storage and management. Furthermore, the design itself overlooked the decommissioning part, which is currently addressed case-by-case based on the building to clean up. We refer to these operations as Post Operational Clean Out (POCO).

POCO deals with several aspects of the dismantling process ranging from the disposition of liquids passing through pipes to the size-reduction of radioactive nuclear wastes. A complex grid of vessels runs throughout the entire building to bring gases and fluids across the nuclear power plant, posing a significant challenge to technicians accessing certain zones. There, narrow doors and tight passages are the only means connecting some of the site areas.

Fluids passing through these pipes when the site was functional demand particular treatments. The most common residue is the fuel slug used for producing the heat in the fission process, and chemical engineers are actively studying how to dispose of these radioactive materials.

A second large duty of POCO is waste management. More than 5 million tons of legacy nuclear wastes wait for decommissioning in the UK, [Woo20]. Since the 1950s, the principal technique for dealing with nuclear wastes has been filling up cylindrical barrels and then storing those containers into buildings for shielding off the radiation. Because of the high demand, operators performed these procedures hastily, with the drawback of keeping almost no record of each barrel's contents and a less than ideal efficiency in sorting and segregating nuclear wastes.

Today, those barrels containing the 5 million tons of legacy nuclear wastes must undergo a review of their conditions. Throughout the years, cracks or dents may have developed, exposing the container to the risk of leaking toxic liquid into the environment. Once we detect a crack, the container goes through a repackaging process. Technicians extract the barrel from its location (often a drum), cut it open with hydraulic shears, and sort the contents based on radiation intensity and size. These critical operations allow the safe storage of legacy wastes, but they also require complex manipulations in extreme conditions.

Operators accessing radioactive cells require to wear an air-fed plastic suit and several pairs of gloves, Fig. 1.1. These precautional measures shield the human being from radiations but encumber their mobility and dexterity.

These working conditions are overbearing for the technicians, who take a total of 2 hours for wearing and taking off the air-fed plastic suit. Moreover, they constantly live with the danger of ripping off the plastic film and exposing themselves to radiations. If they pierce one of the gloves while working, alpha-radiation would penetrate the suit and reach the operator, which would seriously endanger the worker's health.

A secondary source of waste comes from the decommissioning of legacy nuclear wastes. Once exposed to alpha-contaminated environments, clothes worn by operators *i.e.* gloves become new radioactive waste, making a stable incoming flow of new waste barrels every day.

However, human workers are the first source of labour for alpha-contaminated management. As we aforementioned, this type of radiations does not pass through plastic, and operators may carry out the dismantling and size-reduction of these objects by wearing air-fed plastic suits.

On the other hand, facilities with high gamma radiation environments are lethal for humans. In Fukushima's disaster, most facilities are impossible to access directly. The partial fusion of the reactor's core exposed the surroundings to high gamma radiations, which will take hundreds of years to disperse and make challenging the containment and decommissioning problem without endangering human lives.

Robots offer the perfect solution to address the handling of nuclear wastes. Thanks to their resiliency against radiation, the capability of working for hours, and exerting forces greater than humans, robots are the ideal candidate for these tasks. However, the unstructured nature of extreme environments, *i.e.* nuclear sites or disaster response scenarios, makes challenging the deployment of robotic solutions. While manipulators work well where information about the surroundings is available, they often struggle to overcome uncertainties and disturbances from outside sources.

A large body of work has proposed to mitigate these challenges by putting humans in the loop. Teleoperated systems split the robot control into two ends, a master and a slave. The operator pilots the robot using a master device from a safe area, and the robot performs the task on the field. In this master/slave paradigm, the user is in charge of carrying out the task by moving the robot in the necessary positions, which allows for overcomings gaps where autonomous controllers may encounter issues.

Only a handful of nuclear applications have already adopted the teleoperation framework, but they usually rely on mechanical manipulators, which show no intelligence whatsoever and are complex to control. Most of these devices do not mount encoders, and reasoning about inverse kinematics is entrusted only to skilful operators.

In 2016, researchers of the ERL lab collaborating with the National Nuclear Laboratory (NNL) performed the first teleoperated laser cutting of nuclear waste within a nuclear facility using the LaserSnake2 robot manufactured by OC Robotics. The robot is a long reach snake-arm equipped with a lightweight laser cutting head, and its dexterity allows accessing narrow areas that would be difficult to reach either manually or with other robotic solutions. A vision-guided demonstration on a double-walled stainless steel vessel showed that these robots would be a great asset to expedite the decommissioning process. Despite this remarkable result, there is a long road ahead before these conservative industries entirely accept and employ robotic solutions.

The principal downside of teleoperation is that the cognitive load on operators remains. CCTV systems are the only viable option for perceiving the cell from outside, but cameras do not work well under radiational fields. Commonly, they stop working after the first exposure, and the functionalities partially are restored after a few hours. Therefore, technicians resort to tiny lead-glass windows 160 cm tick, which allows shielding from radiation, but dramatically distorts the image. The continuous online compensation of these problems is challenging and creates high stress on the operator's mind.

Autonomous solutions would allow lifting the burden from human workers entirely, who would move to a supervision capacity. However, the planning and control of robotic systems in unstructured environments are open questions. Once a manipulator interacts with an arbitrary surface, a new kinematic chain is created, and the dynamic behaviours change based on the arm and unknown object dynamics.

Researchers are actively studying how to tackle this issue, but no general approach exists yet. Moreover, the problem becomes even more challenging when the environment presents uncertainties affecting sensors measurements. In this thesis, we propose to address these issues at the planning stage, and we introduce robust controllers for executing complex tasks under the vision and sensory disturbances.

1.2 Robotic handling of object in extreme environments

The size-reducing of either barrels or debris and the cutting open of barrels for inspection purposes are challenging tasks that embed most of the problems, that a human worker must tackle, Fig. 1.2.

Once the operator selects a container for inspection, they pull it off from the drum using cranes and use a manipulator with clamps to secure the cylinder. Then, the barrel is moved to a workbench using teleoperated commands. Here, a second teleoperated arm is waiting to carry on the cutting task of the object. After the successful execution, the emptied container is put aside, and the content lies on the workbench. Thus, the sort and segregation task begins, and objects that need further resection are size-reduced one by one. Before moving to the next barrel, the arms store each resized object into the most suitable container. Considerations on radiation, size, material, etc. drive this choice, so that more efficient and cost-effective management of the wastes may be achieved. Every time that a new barrel is full, the technician seals a lid on top. Similarly, the dismantling of walls or pipes follows the same procedure. However, we often cannot move the object on a workbench, and the operator must carry out the cutting on the spot with a rotatory tool, or laser cutter.

Although human workers attain remarkable results, the burden on the technicians is high. The working conditions are far from ideal, cameras work poorly in radiational fields, and operators rely mostly on visual inspection by tiny windows 10 cm thick. Although the window shields from the radiations, its distortion makes it impossible to obtain a good understanding of the scene. Hence, yielding a high cognitive load for the operators throughout the task.

Because of these issues, the pace for decommissioning is far too slow, and at this rate, we would reach the cleanout in hundreds of years.

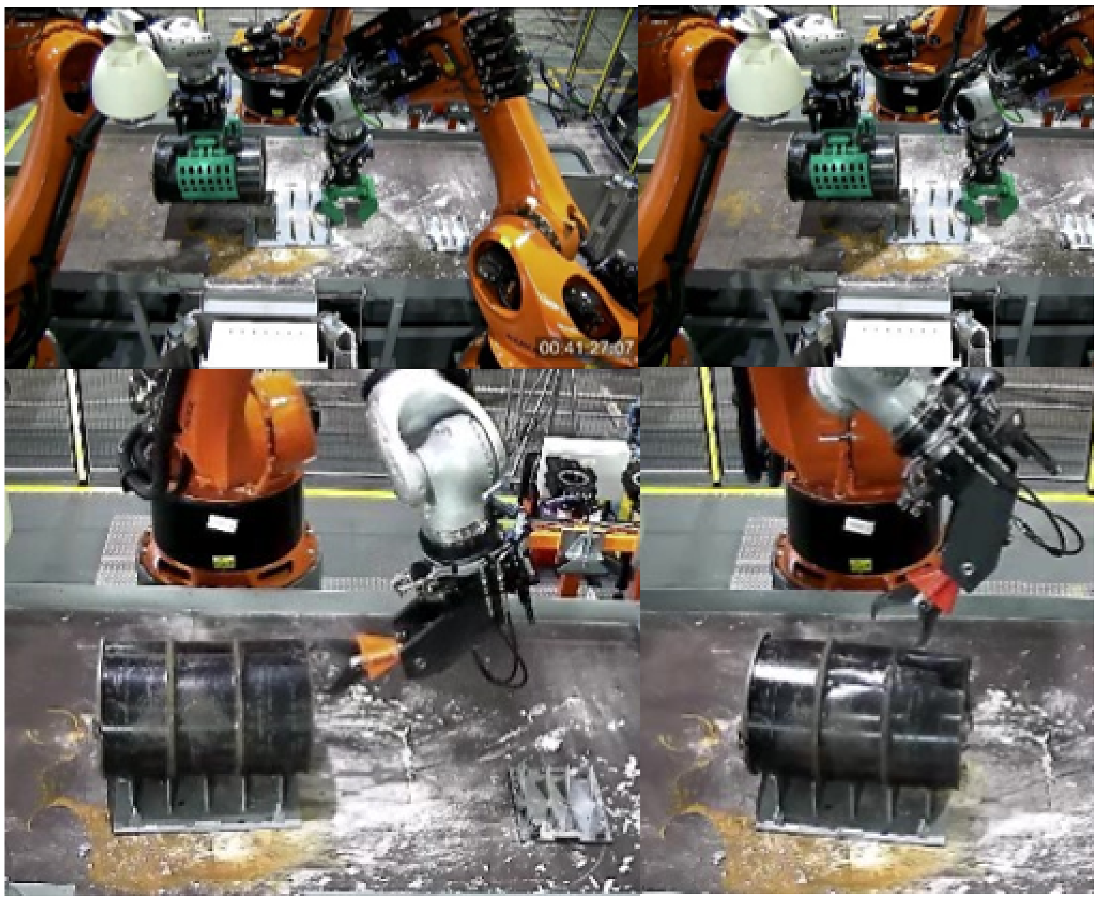


Figure 1.2: A pair of large tele-operated robot arms being used for handling legacy nuclear waste in the UK. The two arms work together to stabilise an old nuclear waste container, and a robot uses powerful hydraulic shears to cut open the container for inspection before repackaging into safer modern containers. Lower:

1.3 Challenges and limitations

Robots might expedite the carrying out of the decommissioning task. They can either execute the operation in entire automation or guide the human workers to achieve better results with less effort.

However, the deployment of autonomous solutions shares some of the challenges with human workers and poses entirely new questions. Generally, a robotic system perceives the surrounding via a vision system *e.g.* 2D or 3D camera, radar, ViCon, etc., and takes decisions based on these data. Sensors can improve the understanding of the environment by also merging signals from different sources. An alternative is to employ tactile sensors, which give the robot an additional degree of information about the manipulated object.

Unfortunately, electronic devices behave oddly under a radiational field. Cameras lying in radioactive chambers have shown to stop working for the first minutes after exposure and then resume functioning, despite a few dead areas in the visual field. As such, measurements are affected by high noise, and we must roll out methods to reject unwanted components and retrieve the original signal.

A more subtle but critical issue with irradiated objects is that radiations might change the material's characteristics. For instance, an aluminium surface may have higher friction than usual, or a block of concrete may be softer than it supposes to be. The combination of defective sensor readings and lack of prior information makes the interaction robot-environment extremely challenging. Generally, compliant controllers achieve acceptable performance by leveraging the ideal model of the object's characteristics and adjusting the misalignments with the online readings. On the other hand, these approaches are not straightforward viable in hazardous scenarios, and we must consider more robust methodologies.

In this thesis, we propose to exploit prior knowledge of the task to attain better results throughout the execution. We reach robust manipulation operations against noise on sensors by relying on information obtained beforhand.

1.4 Contributions of this thesis

In this thesis, we tackle the problem of handling objects in hazardous scenarios. As we have discussed, the problem is rich and links several open research questions.

The cutting open of a barrel demands the robot to plan and carry on a force-full task on the object's surface. The container must hold still using clamps while the arm cuts open or size-reduces the object using a tool, *e.g.* rotary saw. The extraction process of a cracked barrel from a drum needs to select the best possible grasping candidate such that we can pull off the barrel safely.

Disturbances and uncertainties make the environment unfavourable, and robots must show robust behaviours no matter the noise they encounter. While some efforts go towards developing more robust hardware platforms capable of rejecting the disturbances, we propose to use prior knowledge about the task to improve the success rate throughout the execution.

If the robot must interact with an object within a nuclear power plant, sensor misreadings and uncertainties in the object's properties endanger the task's success execution.

As been observed that objects exposed for long periods to radiation might change characteristics drastically, *e.g.* friction, stiffness, durability, etc. These effects diminish the pool of prior knowledge that we can use, and consequently, the robot is prone to obtain inefficient results or even fail its execution. If we set off the robot to cut a sheet of aluminium considering the wrong properties of the material, the robot may stop working because we have underestimated the necessary forces, and the prescribed force is not enough to fulfil the task.

Because these uncertainties in sensing and material properties are impossible to estimate beforehand, we need to provide the robot with robust planning and control.

The manipulability value connects the robot configuration with the possibility of exerting forces (or applying velocities) at the end-effector. If we generate a path with high manipulability throughout, the robot will be capable of handling unexpected online forces requirements. Therefore, if we underestimate the necessary forces to cut the aluminium sheet, the robot can leverage the extra possible forces to reach the destination. Contrary, paths with low manipulability reduce the robustness throughout the execution, and the unexpected forces requirements may overcome the robot available effort.

Because of the design, a robot has a few unalterable characteristics. The combination of joints and links describes a unique kinematic chain, which (in most cases) stays the same for the entire lifetime of the manipulator. Moreover, the mechanical properties of links are unique, *e.g.* length, size, and weight do neither change with time nor under radiations. Therefore they may represent the cornerstone of strategies inherently robust to uncertainties.

The most common metric for assessing the robot's capability, using just kinematic properties and the configuration, is the manipulability. The manipulability describes the residual capability to move when the arm is in a specific configuration (Sec. 3.2.3 for more details).

Leveraging this index, we propose to plan a path for the robot that maximises the manipulability throughout the task. Such a path would improve the likelihood of successfully carrying out the operation, despite the uncertainties during the execution. Disturbances are hardly foreseeable, and a path at high manipulability would grant the capability to depart from the commanded trajectory and overcome unexpected behaviours, *e.g.* a thicker layer of metal while cutting a plate.

The second most important element for handling an object is to perceive its position and shape. Industrial methods rely on CAD models of the objects on which we want to operate. They often have mm accuracy, and the model provides perfect knowledge of its properties using mechanical analysis, *i.e.* ADAM or FEM. However, we miss this information about nuclear waste. Due to radiation exposure, corrosive solutions inside the barrels and the rough dismantling in the '50s, objects pulled off by the containers do not have their original shape. We then must scan those items one by one using 3D cameras and create a collection of views describing the object (a Point Cloud). Although we deploy high precision sensors, radiation intensity jumbles the readings, and the final result is a noisy representation of the initial object.

Thereby, we propose to assume information about the object we want to operate. We consider objects (or portions of the object), which shape a Riemannian surface (Sec. 3.2.4 for more details). Thanks to this hypothesis, the surface upon which we operate is smooth, and we exploit the Riemannian Geometry to generate points on the real surface even though we work with noisy Point Clouds.

We integrated the manipulability and the Riemannian surface concept in a sample-based path planner called RRT*-RMM. This algorithm takes three inputs a noisy Point Cloud and an initial and goal points. And it generates a path with maximum manipulability that connects the two points and constrains the end-effector of the robot to always stay in touch with the object's surface. Although we select the common RRT* framework for developing our algorithm, the introduced methods are extendable to any sample-based path planner.

RRT*-RMM has been designed assuming the arm has a fixed-base. Therefore, it falls short when the only admissible path connecting the two points is outside the reachable space, *e.g.* a u-pipe, or while cutting a large object that is too close to the robot.

We, therefore, propose an extension to our RRT*-RMM called RRT*-CRMM. This new method accounts for this problem and generates paths for mobile manipulators instead. Mobile manipulators have a virtually infinite workspace. However, they pose an additional constraint on the motion of the vehicle underneath the arm. In the case of holonomic platforms (the robot can instantaneously move toward any direction), RRT*-RMM may provide admissible paths reshaping the kinematic chain carefully, *e.g.* considering three cartesian joints instead of the wheels. More challenging is the case of non-holonomic platforms. These robots also can reach any configuration, but the vehicle must comply with the mechanical drive, *e.g.* skew-robots has wheels that work in groups: left and right.

Our RRT*-CRMM accounts for the additional constraints using the redundancy of the robot degrees to achieve paths with greater manipulability, whose end-effector: stay in touch with the surface and drives the base satisfying the non-holonomic structure. We introduce a formulation exploiting the null-space of the robot to obtain complying movements.

Our RRT*-RMM and RRT*-CRMM use a scalar index that retains information about the robot's manipulability. However, the original shape of manipulability is a squared matrix. It is derived from the robot's kinematics and shapes an ellipsoid in the operational-space, centred at the end-effector pose.

The eigenvectors of the matrix describe the potential remaining movements along any direction in the operational-space. This definition allows releasing the concept of manipulability from the robot. Instead, we may attach it to the task we want to perform in operational-space. In this regard, the manipulability matrix becomes robot agnostic, and we describe every task by a set of template matrices. For instance, the vertical peg in the hole task requires a matrix with a great sensibility to the change of force in the z-axis and low mobility in the plane parallel to the hole.

The beneficial effect of designing the manipulability matrix in operational space is twofold: it allows naive users to quickly set up the optimisation process and allows harnessing technicians' high expertise by embedding their knowledge in the planning framework. Thereby, it is straightforward to tailor the optimisation to the application at hand.

We propose an optimisation-based algorithm (called TEMPO), which takes the target manipulability matrix as input, and returns a trajectory for the robot matching the matrix. The distance between the current manipulability matrix and the target matrix is computed using a particular divergence metric. It exploits the symmetric property of the manipulability matrix and is fast to compute. Because of the algorithm's nature, we propose a close-formed method to calculate the divergence's gradient.

Moreover, the tool used for cutting plays a critical role in assessing the importance of smoothness throughout the path. While some tools, *i.e.* laser cutters, accept jerky paths, others, *i.e.* rotatory saws or hydraulic scissors, must follow a continuous trajectory to reach the maximum efficiency while cutting. TEMPO further improves RRT*-RMM and RRT*-CRMM and includes a second objective function, which accounts for the end-effector trajectory's smoothness. Although we do not assume obstacles in the TEMPO formulation, we show how it is straightforward to include further objectives already present in the literature.

Once we have a trajectory for our robot, we must entrust a controller to carry out the operation. However, the problem of controlling a robot that must interact with the environment is challenging. Even more so, when the task involves full-force actions such as cutting open a barrel.

Canonical approaches leverage the hybrid force/motion control theory. Because the robot cannot apply force in the same direction of its movement, these methods cast the two actions in orthogonal space using selection matrices. Here, we exploit this concept to design a controller that can leverage the improved manipulability and overcome the possible uncertainties throughout the task execution. Our hybrid force/motion controller acts as a canonical PID in the force space and shows compliant behaviours in the motion space. The additional capability to move helps the robot adjust its movements at runtime and successfully execute the cutting operation, even when the demanded effort is greater than the initially estimated.

Until now, we looked at how to plan and control a robust interaction. However, the automatisation of the process requires the robot to grasp the barrel and react to unforeseen forces throughout the resizing. In this thesis, we do not investigate the grasping generation problem, and we will usually assume that a collection of grasping candidates is provided by one of the state-of-the-art algorithms in the literature. We instead pivot our research on the selection of the best grasping candidates for attaining a particular task.

The knowledge of the task provides a large source of information, and we can exploit those notions to improve the success against the uncertainties of the environment. The usual ranking of grasping candidates based on stability metrics, like force closure, can be further improved by means of a secondary objective.

We introduce a metric based on the distances from obstacles that permit the re-ranking of the present set of grasping candidates. The newly generated grasping point will permit the robot to pick up the object and perform the task without collisions. We then may compute the trajectories for the robot beforehand and avoid complex re-manipulations and dangerous configurations.

Other authors pursue the same strategy using several metrics, *e.g.* manipulability, force, safety, etc. Therefore, we investigated how to solve the multi-objective problem of selecting the grasp that best balances the objectives. We picked four metrics: manipulability, force, safety, and collision avoidance; and we show how the weights provide a valuable method for prioritising the objectives, *e.g.* collision avoidance is more important than the manipulability of the robot.

The cutting open of barrels is a bi-manual task. One arm must grasp and hold the object still, and the other must exert forces on the object. In scenarios with a few uncertainties, we may design the grasping tool so that we always reach a secure grasp. On the contrary, the design of bespoke tools is not enough when we have scarce information on the environment and properties may have changed from the last observation. Therefore, although the grasp analysis displays the grasp being force closure, the objects may still tilt or rotate in-hand, *e.g.* the real friction is less than estimated.

We then must come up with a method for overcoming the unforeseen uncertainties. Although the problem is challenging, we propose to leverage the information that we have of the cutting path to select the best grasping candidate among a set of given configurations.

Also, we propose an integral adapting sliding mode controller for the cutting arm that displays robust behaviours against disturbances and mismodelled dynamics.

Thanks to this asynchronous strategy, the two arms may carry out the cutting task and avoid unwanted slippages of the object throughout the execution.

1.5 Publications arising from this thesis

The work developed in this thesis rests upon a foundation of peer-reviewed contributions and some unpublished material. We list below the published papers associated with all authors involved. A brief description bridges each contribution to a chapter.

As for the published material:

- **T. Pardi**, V. Ortenzi, C. Fairbairn, T. Pipe, A. M. Ghalamzan E. and R. Stolkin, "*Planning Maximum-Manipulability Cutting Paths*", in IEEE Robotics and Automation Letters, vol. 5, no. 2, pp. 1999-2006, April 2020, DOI: 10.1109/LRA.2020.2970949
  - T.P. was responsible for formalising the theory, the planning algorithm, and provided the results. V.O., A.G. and T.Pipe, provided advice on the theory. C.F. provided industrial insight on the application. R.S. provided advice on general theory and application. Chapter 3 describes the mathematical formulation and the experimental section of the manipulability-informed path planner.
- **T. Pardi**, V. Maddali, V. Ortenzi, R. Stolkin and N. Marturi, "*Path planning for mobile manipulator robots under non-holonomic and task constraints*", 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 2020, pp. 6749-6756, DOI: 10.1109/IROS45743.2020.9340760
  - T.P. was responsible for formalising the theory, the planning algorithm, and provided the results. V.M. actualised the produced paths to the VREP simulation environment. V.O. and N.M. provided advice on the theory and the writing. R.S. provided advice on general theory and application. Chapter 4 extends the previous paper to floating-base robots and accounts for non-holonomic constraints.
- **T. Pardi**, R. Stolkin and A. M. Ghalamzan E., "*Choosing Grasps to Enable Collision-Free Post-Grasp Manipulations*", 2018 IEEE-RAS 18th International Conference on

Humanoid Robots (Humanoids), Beijing, China, 2018, pp. 299-305, DOI: 10.1109/HUMANOIDS.2018.8625027

  - T.P. was responsible for formalising the theory and the model and provided the results. A.G. provided advice on the theory and the writing. R.S. provided advice on general theory and application. Chapter 6 describes the mathematical formulation and the experimental analyses of our task-informed grasp selection algorithm.

- **T. Pardi**, A. M. Ghalamzam E., V. Ortenzi and R. Stolkin, "*Optimal grasp selection, and control for stabilising a grasped object, with respect to slippage and external forces*", 2020 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids), Munich, Germany, 2020

  - T.P. was responsible for formalising the problem, carrying out the experiments, writing the manuscript, and analysing the results. A.G. and V.O. provided advice and guidance on the idea and the writing. R.S. provided advice on general theory and scope for applications. Chapter 7 explains the mathematical formulation and the context within our method works. It also reports experimental analyses of our adaptive robust approach using a real robot.

As for the unpublished material:

- we presented the work in Sec. 6.3 to the peer-reviewed *Task-Informed Grasping (TIG) for rigid and deformable object manipulation* workshop at the IEEE-RSJ International Conference on Intelligent Robots and Systems, 2018

  - "Selecting the best grasp during post-grasp manipulation under multi-objective criterion" - **T. Pardi**, R. Stolkin, A. M. G. Esfahani

- Chapter 5 proposes an optimisation-based trajectory planner, which is in progress for being submitted to IEEE Transactions on Robotics (T-RO).

1.6 Structure of this thesis

This thesis is structured as follows. The second chapter presents the work related to our research and shows the difference between the state-of-the-art methods and our contributions.

Chapter 3 focuses on planning movements for a robot tasked with cutting nuclear waste. We present the mathematical formulation for solving the problem, which exploits the robot manipulability and Riemannian geometry. We endow the versatile sample-based framework RRT* with our improvements, and we present the novel RRT*-RMM algorithm. The outline of the algorithm is thoroughly analysed. We show the effectiveness of our method on real tests, using a fixed-based 7 dof panda robot.

Chapter 4 broadens the functionalities of our RRT*-RMM and extends it to floating-base platforms. In particular, we deal with the challenge of planning for non-holonomic mobile manipulators. Because of the additional constraints on the instantaneous robot movements, we introduce a technique for planning admissible paths using the Null-space of the robot's kinematic. The novel version of the algorithm is called RRT*-CRMM. We analyse the newly added lines of the algorithm with respect to RRT*-RMM. We test our RRT*-CRMM on a collection of Point Clouds with low and high resolution. A simulated robot validates our approach using a skew-steering robot, Husky + HR5 manipulator.

In Chapter 5, we present the path planner TEMPO. It is an optimisation-based algorithm accounting for the smoothness and manipulability matrix throughout the path. We formulate the problem using the Riemannian space of Positive Definite Matrices, and we employ a fast metric to compute the distance between two matrices. An extensive set of experiments is presented using Point Clouds with various shapes, sizes, and resolutions. Moreover, we introduce a hybrid force/motion controller with compliance behaviours on the motion space.

Chapter 6 investigates the grasp selection problem. First, we propose a method for choosing the best grasp location based on a novel collision avoidance metric. Then, we study the same problem when it is subject to multi-objective criteria.

In Chapter 7, we research the problem of performing the cutting task using dual-arm solutions. We present a grasp selection algorithm based on actuators' strain, and we propose a robust controller that accounts for the uncertainties in the system. Our controller deals with model uncertainties and allows movement of the arm matching the estimation of the friction between object and gripper. We show results in the particularly challenging problem of 2-D object manipulation, where gravity is not present. We carry out an experiment on a real robot. For experimental purposes, we use a 7 dof Panda robot as one arm, and a human operator acts as the second arm.

Finally, we draw the conclusion of our work, and we discuss possible outlooks for the future.

CHAPTER 2

Literature review

Robotic cutting and handling of objects engender a challenging problem of path-planning, control, and grasping. The arm must either grasp an object for performing the following operation *i.e.* drive a screw or dispose of nuclear waste, or resize an object using sophisticated tools *i.e.* rotatory tools or hydraulic shears. In industry, many applications, *e.g.* the large production of metal components, demand millimetre accuracy, and they often use machines like the CNC machine (Computer Numerical Control) to provide high precision and repeatability. Those approaches work best in structure environments, where engineers may design the pieces beforehand, and the machine has perfect knowledge of the model. On the other hand, hazardous environments fall short very quickly of reliable models, and we must come up with more robust and efficient methodologies to cope with the uncertainties.

Although the data collection is noisy in outdoor scenarios or radioactive cells, these applications do not demand following a path with high precision. The focus of the task is usually resizing nuclear wastes or removing debris to rescue people. Thereby, a departure from the original path is positively accepted since it improves the chances of task success.

This *rough* cutting is critical for using robots for cleanup of legacy nuclear waste [MRR+17, TGT+16].

2.1 Path planners

In the past 30 years, researchers have extensively studied the problem of planning a path for robots. Two main classes have emerged: sample-based and optimisation-based planning.

2.1.1 Sample-based planning

Sample-based planners are fast and efficient iterative methods which generate a path using the space of configurations of the robot. At every iteration, they generally do two operations: they randomly pick a point within the set; they connect the new point to the already evaluated points. Sample-based planners store every point in a tree structure, which yields logarithmic complexity to add new elements and search within the tree, [LaV06].

Many authors have proposed algorithms that belong to this class. [KSLO96, AW96] proposed the Probabilistic Roadmap (PRM), which generates paths connecting an initial and goal point and avoiding obstacles. After sampling a new configuration, the algorithm checks whether the robot is in a collision or not and acts accordingly. The new configuration becomes connected with the nearby elements, and the local planner attempts to bridge the two nearby points.

Arguably the most common sample-based approach is the Rapidly-exploring Random Tree (RRT) [VZ11]. It employs a KD-tree structure for efficiently addressing high-dimensional problems. It quickly obtains collision-free paths connecting two points and provides a framework for planning in many scenarios.

However, RRT generates only sub-optimal trajectories. In 2011, [KF11] proposed the optimised version of RRT, called RRT*. They introduced a further step, called *rewiring*, which re-evaluates points in the tree based on the new information. It is proven to reach the optimal path when the number of iterations goes to infinity. This new type of algorithm is a *anymotion* planner. It quickly generates a feasible path and improves the result further with more iterations.

Improvements to RRT and RRT* are found in [KL00, KWP$^+$11, SH16, WM20]. RRT-Connect, [KL00], is an efficient single-query path planner. It exploits the idea of RRT by running two parallel algorithms, one from the start and one from the goal. As soon as the trees touch each other, a feasible path becomes available. [KWP$^+$11] extended RRT* to employ more efficiently the time after the initial path for any time algorithms. [SH16] proposes a generalisation of RRT and RRT* called Lower Bound Tree-RRT (LBT-RRT). It aims to generate the shortest path between two points and uses an approximation factor as a metric. The algorithm is proved to converge within the $1 + \epsilon$ of the optimal solution. The ϵ drives the approximation level. This approach generates high-quality anytime motions that can be tailored to the application at hand. Furthermore, [WM20] proposes to sample the configuration space non-uniformly and focus areas where the optimal path may exist. The methodology accelerates the path planning process and results in higher efficiency. To drive the sampling, a map of the environment is generated using the Voronoi graph technique and a heuristic accounting for the length of the path and distance between obstacles. A probabilistic approach drawn by the particle filter context is introduced by [MS07]. Rather than building a tree by sampling a point at every iteration, the authors propose a stochastic approach, where every point is simulated multiple times before being considered. It allows the generation of paths that cope with uncertainties and are probabilistically safer despite the little knowledge about the environment.

[Wv13, LCLX18] propose a kinodynamic version of RRT*. The former optimally generates paths for systems with linear dynamics. A controller with fixed-final-state and free-final-time addresses the connection between two points and yields paths feasible by any linear system. Moreover, the latter tackles the problem of planning motions for systems with non-linear kinodynamic constraints. To estimate the cost function between two configurations, they deploy a Neural Network (NN), which predicts the cost. The combination of the two approaches yields a near-optimal motion planner for complex constrained systems.

A body of the literature studies the use of sample-based approaches for driving vehicles, [KFT$^+$08, RMSL11, WGY$^+$20]. [KFT$^+$08] shows with many simulations how simple and yet effective the RRT algorithm can be against unstable dynamics, significant drifts, and noisy limited sensing. They applied these results in the context of the MIT DARPA Urban Challenge vehicle. Mobile vehicles are often the preferred choice for service robots. Because they move around the environment and interact with people, it is paramount that these robots are aware of social conventions while traversing. [RMSL11] proposes the Risk-RRT, which generates navigation strategies that accounts not only for people as obstacles but for the interaction between subjects. By avoiding cross people talking, the human-robot interaction is perceived as more natural and forthcoming. Other work, like [WGY$^+$20], aim to improve the performance of the anytime algorithm. Their proposed R-RRT* accounts for the speed, memory allocation, and large sampling space by using three strategies. They introduce a connected domain sampling, an elliptic modelling sampling and a path optimisation strategy for this purpose.

Another instance of the application that is suitable for the sample-based algorithm is the combination of manipulation of an object to achieve a task. In [ZSKW12], the authors exploit the RRT flexibility to design an algorithm for push manipulation operations. The method explores possible pushing actions and develops a map. The map is then used by a local push planner that uses predictive models.

RRT-like algorithms cross the robotic domain as well and find applications in bio-engineering and biology. In [CJS07], the problem of computing pathways for a ligand to exit from the active site of a protein is tackled using RRT. The problem is at a high-dimension, and RRT shows to be an efficient method for obtaining feasible paths. Also, [JLYS11] tackles the critical tasks of cell sorting and cell fusion in the drug industry. A pair of optical tweezers must trap a cell and move it toward the target position. To generate a collision-free path, the author employs the RRT as a path generator.

A survey on RRT can be found in [NKH16].

2.1.2 Optimisation-based planning

Optimisation-based planners apply a more canonical method to planning. An objective function embeds the goal of the path planner, and the gradient drives the path toward the optimal path. These methods have the beneficial effect of generating ready-to-execute trajectories. In contrast, sample-based methods require a local controller or a filter to bridge the waypoints more smoothly.

Early work by Khatib modelled obstacles as Artificial Potential Fields (APF), and optimised collision-free paths by descending an energy gradient, [Kha86]. The definition of the artificial potential field is not unique, and several authors have proposed many formulations *i.e.* [HA92, DB08].

One shortcoming of APFs is that local minima can trap the robot before reaching the goal. In [MJM01], the authors proposed a method based on simulated annealing to escape local minima and obtain a globally optimum path.

More recent methods for path-planning by gradient descent of cost functions consider the path as a function and optimise the entire trajectory at every step. Well-known algorithms that carry out this approach are CHOMP [RZBS09], STOMP [KCT$^+$11], and ITOMP [PPM12].

The Covariant Hamiltonian Optimisation Motion Planner (CHOMP) takes a trajectory as input and returns a new trajectory that minimises the objective functionals. Instead of considering one point at a time, the algorithm provides a gradient step for every point in the trajectory at every iteration. Furthermore, to always obtain smooth movements from one path to another, they propose to project the free-gradient on the Riemannian manifold of smooth trajectories. The main objective of the functional is the path length, but the authors introduced a method to consider also obstacles and robotic constraints. Although CHOMP shows good results on a variety of robots, it is time-consuming, and other methods have improved the efficiency. The Stochastic Trajectory Optimisation for Motion Planning (STOMP) [KCT$^+$11] uses stochastic gradients. It has the two-folded contribution of addressing cases where the objective functional is non-derivable and reduceing the time consumed to carry out the optimisation process. Moreover, the authors accounted for a further element in the objective function, the torque at every joint throughout the motion. Incremental Trajectory Optimisation for Motion Planning (ITOMP) [PPM12] tackles dynamic environments. Because the collision may happen after the algorithm has planned a trajectory, the planning and execution phases are carried out in an iterative fashion. This approach allows the robot to avoid moving obstacles in the scene without any prior knowledge.

Optimised-based methods are versatile. For instance, [JH19] proposes a whole-body motion planner for walking excavators. The walking excavators are complex floating-based robots that counts 31 dofs, and they must undertake several limitations in term of effort, velocity, and range of action. Alongside the optimisation process, the authors propose a spline function that connects all points and generates smooth trajectories.

While executing a task two or more costs of the objective function may disagree. Therefore, a trade-off methodology must be in place to weigh every cost. However, not all the elements have the same priority, *e.g.* collision avoidance is more important than minimising the torque effort. In [GR17], the authors leverage genetic algorithms and projection to prioritise the costs in the objective function and obtain computationally efficient trajectories.

[AJM$^+$18] integrates offline learning and real-time motion planning using Bayesian optimisation. First of all, they create a database of end-slope cubic Bezier functions that are optimised using Bayesian optimisation for a set of points. At run-time, the robot seeks the closest trajectory using the k-Nearest Neighbour method. The extracted trajectory passes through a new optimisation using available points, and it is finally carried out by the robot. Thanks to the stored data, the algorithm reaches quick reactive motions in the tracking error task.

The optimisation-based motion planner extends also to multi-robotic systems, [GFC18]. The handling of compliant parts is a challenging operation. Generally, more than one robot must operate on the same object for bending in the shape of metal sheets. Based on the deformation, an objective function is designed to achieve the target performance.

A recent survey on optimisation-based motion planners can be found here [YPW19].

2.1.3 Informed planning

The current literature predominantly focuses on avoiding obstacles and minimising the journey length, [KRJS17]. However, when the task is more complex than connecting two points avoiding obstacles, *i.e.* interacting with the environment, maximising the safety in a shared environment, or minimising the strain of movements, canonical path planners do not provide reasonable solutions.

Various authors have sought to augment the classical path-planning approaches by incorporating modified cost functions, based on additional information, to induce useful supplementary robotic behaviours. A cost-based optimisation approach was proposed in [KSBB07] to enable an Autonomous Underwater Vehicle to plan a path between the specified start and destination locations. The robot plans large deviations in its route to avoid adverse currents while exploiting currents in useful directions to minimise energy expenditure during the journey. Related Unmanned Aerial Vehicle literature also considers environmental factors, *e.g.* [NVTK03]; and planning on non-flat surfaces with constraints is considered in legged locomotion [LB18].

A few works propose to heuristically biased the sampling, *e.g.* [US03], whilst [GSB14] proposes an Informed-RRT*, which obtains efficient results by biasing the sampling toward directions more likely. Other approaches drive the RRT search via prior knowledge of the robot. [BKDA06] leverages on the inverse kinematics to generate feasible sequences of configurations, and [LH03] guides the sampling using the manipulability of the robot. In [JCS08] and [JCS10], a transition-based RRT algorithm, driven by a work-based cost, generates a collision-free path between points. They propose an energy function, which leads the path planner toward areas with lesser cost. A method for escaping local minima in the map is proposed using a momentum-like strategy inspired by mechanical energy.

[RJCC17] and [JRCC0] tackled the same problem but exploited the learning by demonstration approach. They learn to generate paths by mirroring the actions demonstrated by another robot. The algorithm records the manipulability's ellipsoids throughout the motion and generalises to new situations.

In other cases, the task itself limits the robotic movements. Deburring operations or passing over a glass of water are just two examples of strong constraints to the end-effector pose throughout the task execution. If the above approaches are used to plan motions on a surface for a contacting end-effector, the resulting path may pass through regions: that lie outside the robot's reachable workspace; that correspond to singular configurations of the robot; where the robot is unable to exert the required cutting forces.

However, there is a relatively small body of work on motion planning under end-effector constraints. Work as [HA01, BSK11] are fuelled by the need for moving humanoid robots. The high number of degree of freedom makes the problem very challenging, and it demands careful planning and control for achieving good results. In [KUSP16], a sample-based algorithm generates paths on the tangent plane to the surface.

When the task demands the robot to cover a wide range of motions, mobile platforms are the preferred option because they greatly improve the workspace of the robot up to virtually infinite. However, the planning for mobile platforms must tackle the additional constraints of the moving base. A surge in demand for mobile manipulators is recorded in many domains. For instance, in many medical applications, telemanipulators and mobile manipulators assist the surgeon in following pre-defined procedures or scaling the doctor's movements when the operation is performed using the teleoperation approach, [Paj15, PP14b, PP15, PP14a].

While there is work investigating the path planning for Non-holonomic Mobile Manipulators (NMM) [DOG07], it is generally focused on path length. Thereby, the best solution to effectively handle NMMs is still an open question and greatly depends on the application. Recent works like [LHSG$^+$19, SC19, WGL$^+$19] propose to apply multi-objective optimisations while planning for mobile manipulation tasks. They leverage the degrees of freedom of the entire platform to improve its kinematics and dynamics throughout the motion.

2.2 Grasping selection

The planning of intelligent grasping configurations involves many factors, some depend on the object *i.e.* shape, friction, and position with respect to other objects in the environment, others from the gripper we are using *e.g.* number of fingers, mechanical features, dexterity, etc.

A large body of work investigated how to synthesise a grasp configuration. Force-closure [FC92], or form-closure analysis [DLW01] are classical approach to compute stable grasping configurations based on the 3-D model of an object. They consider that every contact finger-object forms a friction cone, where the cone's point is at the contact, and it expands inward on the surface of the object. Thanks to this formulation, an algorithm is able to discriminate whether a set of grasping points holds the object or not.

However, a 3-D model of the object is not always available, and we must resort to Point Clouds. To overcome this issue, many authors have proposed algorithms using templates and probabilistic algorithms. [VDAD10] and [BDN$^+$07] apply a similar method, they store a set of commonly feasible grasps (template) and look for matching areas between the template and current object shape. [KDA$^+$16a] uses kinesthetic demonstrations to learn a probabilistic model. At the running time, the algorithm finds the most likely configuration for the gripper on the observed PC. In [AMO$^+$18], the Local Contact Moment (LoCoMo) algorithm exploits matching geometric features between object and fingers to obtain feasible grasps for parallel jaw grippers. Other works investigate artificial Neural Network (NN) approaches for generating grasp configurations. In [GtPSP16, AtPP17], the authors initially have generated a set of feasible grasping configurations for parallel jaw grippers using canonical metrics. Then, a deep learning network has learnt to generate feasible grasps on PCs. These methods have the benefit to generate grasp configurations also when partial PCs are available. [KSB16] proposes to leverage supervised learning for teaching the robot where to grasp objects from a partial PC. A *ranking loss* leads the network through the learning phase and weighs more those grasps that have positive labels in the dataset.

A parallel line of research moves the focus to the design level. [DH06] introduced a compliant two fingers gripper. The joints are formed by elastomeric flexures, and actuator and sensor components are embedded within rigid polymers. An anthropomorphic soft-hand driven by tendons has been proposed by [CGF+14]. The hand is controlled by a single actuator, which pulls the tendons to close the grip. Because of the wiring between the fingers, the hand imitates the most common closure method carried out by human beings when grasping objects. The phalanges dislocate to wrap the object in a safe and robust manner. Moreover, work as [WTH17] proposed a gripper entirely made of 3-D printable flexible materials. The closure/aperture phases are performed by inflation/deflation of the chambers within each finger. The gripper reaches several configurations and shows great capabilities to grasp very delicate objects, like fruits or bread. In [PPL+20], the authors propose a solution for bin-picking in the industry. They introduced a hybrid solution which merges a soft end-effector with a canonical 3-axis Cartesian robot. Thanks to the soft gripper and wrist, the robot is able to grasp a variety of objects in a cluttered environment.

2.2.1 Task-informed metrics

Although these algorithms propose stable and robust grasps, the pivoting idea is to grasp. However, in many applications, the grasping action is a purposing task. We pick a ladle to stir a soup; we grasp a screwdriver to drive a screw in the wall. This new perspective shed a different light on the grasping problem, which becomes a strategical problem rather than a geometrical one.

A relative little work have investigated this concept in the past, but it is gaining momentum in recent years. The more robots are entering the real world, the more the afterwards task becomes critical.

Work as [AMK$^+$16, MAS$^+$16, MGER17, GEAFRGS17] assume a task is given for the robot, and they propose to select the grasp configuration that better fits the following task among a set of given candidates. [AMK$^+$16] developed an index based on arm manipulability. Every grasp configuration is evaluated using this index while carrying out the trajectory. The algorithm returns the best grasp configuration in terms of manipulability. The same authors have exploited this method for performing teleoperation in [GEAFRGS17], where the human operator was guided using the manipulability index. Teleoperated robots, like the DaVinci, are becoming a valuable resource for medical purposes, and surgeons are relying more and more on those robots to perform procedures with greater accuracy. However, doctors are usually naive to robotics, and we need to devise methods for helping the operator and easing the cognitive load. Therefore, [SGEM$^+$19] proposed to tackle the grasp selection for stitching tissues. The algorithm drives the operator toward the grasp candidate onto the surgical needle, which displays the higher manipulability. Similarly, [PKM$^+$20] introduced a teleoperation algorithm that drives the user toward a grasp. However, the strategy accounts for collision avoidance throughout the task. They use a master/slave framework using two robots. The slave mirrors the master's movements, and the master provides cues to the user using force feedback. Although the operator stays in charge of the motion, the robot suggests the collision-free grasp in a haptic-fashion. On the other hand, [MAS$^+$16, MGER17] accounted for the inertia of the grasped object throughout the task. The trajectory yielding lesser inertia is safer and potentially less harmful for a human operator that shares the same working space.

In [DPM17], an approach is presented that exploits a semantic and geometric understanding of a scene for task-oriented grasping. The concept of affordances plays a key role here. Many objects suggest the grasp point by the combination of the shape and task, *e.g.*

to pound a nail, you would take the hammer from the handle and swing the head to hit the nail [ZHL+17]. After selecting an object for grasping, [DPM17] searches affordances on the surface of the objects, based on the task to come, and picks the better grasp configuration. The same concept extends to the domain of Human-Robot Interaction (HRI). Services robots must help the human companion by passing and receiving objects. However, the way a robot picks an object changes the possible grasp points for the human. Therefore, the operator might need to re-manipulate the grasped object before using it [COCC19, OCP+20]. The robot should adjust the grasping strategy to accommodate the human rather than the opposite.

2.3 Robot controllers

Once we selected a grasp candidate and a suitable trajectory, the control law must step in to carry out the task.

Since the 70s' the robotics community has investigated the problem of controlling robotic systems. The effort produced a large body of work that touched many aspects of the robot behaviours and extended to several domains.

Industrial applications deal with two (often opposing) problems efficiency and safety.

In deburring operations, a robot uses a tool to remove burrs from a workpiece. The device must follow the object's profile and brush off all the dirt and debris due to the production process. The forces exerted and the mm accuracy are essential for an efficient deburring in terms of the quality of the polishing and strain on either object or robot.

At the same time, the oncoming Industry 4.0 demands robots and humans to collaborate. More and more factories deploy co-bots that work alongside human operators to improve production. However, the shared space requires the definition of policies such that the technicians are not endangered by robot motions.

Until a few years ago, industrial settings deployed robots only within cages. These zones were prohibited to humans and were providing environments where the robot could easily interact. Robotic arms would perform iterative tasks with high velocities. In [HFY20], the authors propose to use telescopic robots to pick and place objects on a conveyor in order to reach real-time operations.

The main pitfall of these approaches is that they strongly rely on the environment's structure, and most of the motions are hard-wired using teach pendants or manual coding.

More sophisticated methods include the use of neural networks or compliant robots for performing these operations. In [BMK20], the authors use self-supervised learning to teach the robot how to grasp circular objects from a box and insert them into a rod. The vision system predicts the movement of the objects and decides the right action. Although the resulting network performs pick-and-place tasks with great accuracy (around 2.6 mm), it took more than 25000 real experiments to master the task. On the other hand, [PPL+20] endows a 3-axis cartesian robot with a variable stiffness wrist and an underactuated gripper. The robot shows great adaptability to grasp objects of several sizes from Amazon factory-like shelves.

Even more challenging is the problem of making robot-environment interactions. [Pag99] studies tasks that demand this kind of operations, *i.e.* deburring, grinding, chamfering, etc., and defines three phases: free motion, transition, and constrained motion. The latter two are responsible for successful polishing. They designed model-based controllers for each case and switched from one to another based on the distance between the end-effector and surface.

Although they yield good results, the technique undergoes noise in the sensor and is not easy to generalise to other tasks. Therefore, [AD14] tackles the problem with a skill-based method. An expression graph defines the constraints for both robot and task and provides a trajectory for carrying out the operation. This approach explores the creation of flexible strategies that are easy to program, also for robotics-naive users.

When the robot must operate in a scenario with high noise and perform a broad range of tasks, teleoperation comes to the rescue. In the most straightforward case, the operator directly controls the actions of the robot from a base station. However, cognitive stress is high for the user, and mid-ways solutions are usually preferred.

In [RAFGP19], a teleoperated arm is tasked with cutting a soft material using a scalpel. The user drives the robot using a 6-DOF haptic device that provides force feedback that helps the teleoperation. Because the scalpel demands smooth movements along the blade to maximise the cutting operation, the authors propose to constrain the arm to behave like a non-holonomic robot.

Other examples where the deployment of robots have an impact are service robots and the safe&rescue. Non-holonomic mobile manipulators (NMM) are a valuable resource for these domains. In [DOG07], a visual servo controller leads NMMs and exploits the redundant structure to reach non-singular configurations with higher performance.

The problem becomes even more complicated if the robot needs to interact with the environment autonomously. Industrial settings often work in structured environments, and engineers provide the robot with a model of the object beforehand using CAD software. However, miss-alignments between the object and the robot's end-effector might exist, which cause great stress throughout the operation on both parts and may fail in the task. Moreover, in an unstructured environment, we often do not have a priori 3D CAD model, and objects are, at first, perceived using depth sensors, then reconstructed using meshing algorithms like [Hop08].

To overcome the uncertainties between the real object and the perceived model while touching, researchers resort to the class of compliant controllers. These control laws govern the force exerted by the robot on the environment and adjust their behaviour depending on feedback. The compliance may be reached using either *active* or *passive* methods. The former uses motion controllers within the force loop for generating forces that emulate a spring repulsion proportional to the displacement between the current and desired position. The latter mounts elastic components after the end-effector, which passively complies with the touched surface.

Impedance controllers are a class of dynamics control linking force and position. They find the principal applications where manipulators must interact with the environment, and the exerted force must display a particular dynamics behaviour. Once the robot touches an external object, the interaction is governed by a “spring constant” defined within the control law, which scales with the displacement of the end-effector from the reference position.

A large body of work investigates the use of this class with single and multi- arms systems. In [LSB$^+$18], the authors introduce a cartesian impedance control for dual-arm operations. The method projects the inverse dynamics of the arm to attain contacts perpendicular to the touched surface, then an impedance controller allows the lifting and moving of objects with unknown weight and under exogenous disturbances.

Other works address the same problem but leverage the concept of admittance instead of impedance. [GZA18] proposes a framework that balances the forces for each arm using the distribution of the load carried. A PI controller pursues a virtual target for changing the object position. To extend the approach to several tasks, they use Dynamic Movement Primitives (DMPs). DMPs grasp the key features of the demonstrated operations allowing a variety of compliant manipulations.

Compliant controllers find great use in human-robot interaction as well. Safety policies utilise the capability of displaying a specific behaviour of these controllers to reduce the risk to operators’ health. [JHK19] tackles this problem by using the redundancy of the arm. While the robot’s end-effector keeps a given position and orientation, external contacts occur. The manipulator exploits the NULL space of the kinematic chain to adjust the configuration while maintaining the end-effector position.

Another important category of force controllers is the Hybrid Force/Motion (HFM). Control law belonging to this set pivots around the concept that force and motion cannot exist in the same space. The cornerstone of the HFM theory is [Kha88]. The author proposes a unified approach that casts movements and forces into orthogonal spaces. They apply a selection matrix at the end-effector, which allows movements in predefined directions and prevents the overlapping of force and motion commands. Moreover, [KK88] introduces a formal mathematical description of the perpendicular spaces by shaping the problem using pseudo-velocities. The resulting reduced system retains all the features of the original but a lower dimension. The advantage of the approach is a speed-up in simulation time, but with the downside of no uniqueness in defining the pseudo-velocities. Because of the definition, the new variables may lose physical meaning, and the controller designing may become difficult.

Recently, work as [SH15] exploited the concept of passivity for reaching HFM control. The authors propose a passivity-based controller allowing accurate tracking and compliant impedance behaviour using energy tanks. The method shows robustness to a sudden loss of contact and chattering (intermittent switching between contact and no contact to the surface). Energy tanks suffer from the initialisation issue, and the authors propose a strategy based on task energy, which estimates the strain for the robot to perform the task beforehand.

This last work introduces one of the biggest issues for controllers: uncertainties. Uncertainties represent either missing or defective information of one of the system parts, *e.g.* outliers in sensor readings, noise in the perception, wrong parameter identification, etc. The effects of disturbances can be catastrophic for the system when the control design does not consider them. For instance, a system with numerically estimated parameters is theoretically stable, but uncertainties may bring the system overboard to exhibit unstable behaviours.

Robust control theory deals with uncertainties and provides a collection of methods for designing stable controllers despite the missing or altered information. The drawback of these techniques is usually lower performance.

Variable Structure (VS) controllers belong to the robust controllers class. They revolve around the idea of devising more than one controller and then using the most appropriate based on the state-space condition. Sliding Mode Controllers (SMC) are well-known variable structure controllers. SMCs define a surface in the state-space and force the system to stay on the surface using high-frequency control actions. [PYPY18a] carries out a study on an advanced version of SMC called Integral SMC. It tackles the chattering nearby the sliding surface and the initial “reaching” phase of the controller. These methods broaden the stability margin of SMCs by using sensor readings to skip the initial stage of the controller.

When direct feedback from sensors is not available, SMC misses a critical element to track the error. In [KJL18], a dual-arm manipulator is controlled using SMC for performing assembly work. The robot is sensorless, but they estimate the reaction force using a perturbation observer.

A few authors proposed to employ robust controllers to address robot-environment interactions, *e.g.* [GA20, YL12]. Both work pivot around continuous contact between the robot’s end-effector and unknown surface. The presence of uncertainties may cause a loss of contact throughout the task execution. These works consider this effect a disturbance and use a robust observer to estimate the instantaneous surface gradient. Then, the robot accounts for the change and adjusts its configuration. [GA20] introduce an HFM control for robotic manipulators, and [YL12] tackled the same problem for mobile manipulators. [PS12] reports a survey of the state-of-the-art methods. Other authors dive into entirely different approaches to address uncertainties. [MT17] tackles the problem by pre-computed “funnels”. They deploy this method for controlling a fixed-wing aeroplane avoiding obstacles at high speed. A fast planning and control reaction to the new environment is critical in these cases. The robot plans stable trajectories by skimming a library that contains templates of pre-computed movements, where the robot stays within admissible disturbance bounds.

Besides researchers’ endeavour, controlling complex kinematic chains under uncertainties remains a challenging task. Therefore, some authors have investigated the use of underactuated mechanisms. This class of mechanical systems combines canonical actuated motors with passive motors. Generally, they display more robust behaviours against model inaccuracies and fault tolerance. On the other hand, the control theory for the underactuated robot becomes more difficult.

In the literature, we find an extensive study on underactuated controllers [DHW+17, ZZL$^+$18, OS00, NAM06, LIMO03, Spo96], which address both canonical and robust controllers for stabilising the system around an equilibrium point. However, most of the work tailors the proposed solution to the application at hand, *e.g.* VTOL[SR17], Quadrotor[XÖ08], Acrobot [DHW$^+$17].

In [LIMO03], the authors analyse the control of a planar multi-joint arm and prove that the system is non-Small Time Local Controllable (STLC) when the gravity term is missing. The loss of controllability can be partially restored using an iterative steering method, [DMO00]. The technique drives the system to equilibrium in a feedforward fashion using a double-step algorithm: at first, the method sets in place the actuated joints, then a sequence of *contracting* trajectories move the passive joint to the desired position. Although these techniques move the system to the desired locations, they have the shortcoming of not being capable of promptly answering external forces which may perturbate the object *i.e.* via human-robot interaction.

CHAPTER 3

Planning maximum-manipulability cutting paths

3.1 Introduction

Robotic cutting involves an interesting problem of path-planning for a serial arm under semi-closed chain constraints. The end-effector cutting tool is constrained to touch the cutting surface, thereby forming a closed chain at any given time step. However, the cutting surface can be regarded as a manifold upon which the end-effector has *locally* two or three degrees of freedom to move, *e.g.*, cutting with rotary tools. The problem of our interest is *rough* cutting, *e.g.* in robotic demolition in hazardous environments. In such applications, the exact cutting path is not important, as long as the robot successfully, *e.g.*, cuts an object into two pieces, or cuts open a container to inspect its contents. We, therefore, address the cutting-path planning for a serial manipulator.

Rough cutting is a key element for using robots for cleanup of legacy nuclear waste [MRR$^+$17, TGT$^+$16].

Although the variety of tooling and tasks present in nuclear sites, only a few instances of autonomous systems were allowed in the past. [DR20] deployed an autonomous vision-guided robotic laser cutting of contaminated metal inside a radioactive facility. Lasers, and other non-contact methods such as water jet or plasma cutting, are convenient in that no contact forces are exerted, although close geometric surface following (with a few millimetres stand-off) must still be achieved.

In contrast, we are now considering the use of axial rotary cutting tools (similar to a milling machine cutter). Kinematic path-planning constraints for such tools are similar to those for a laser: the cutter axis must be maintained normal to local surface curvature, and rotations of the robot around the tool axis are allowable. However, with the rotary cutter, forceful interactions between the robot and cut materials of uncertain properties introduce significant perturbations. We would like the robot to have sufficient manipulability to provide the capacity for responding compliantly to such perturbations while following a cutting path between two points given by a human operator. Maximising manipulability [SGEM+19] also results in motion plans which will not cause any joints of the robot to pass near singularity configurations. The robot must interact with arbitrarily shaped objects. The lack of 3D models of these objects requires the use of additional sensors, *i.e.* 3D cameras, to perceive the position and shape of the object. In normal conditions, high-end off-the-shelf sensors capture highly accurate readings of the object, and the interpolation of the recorded points produces a precise 3D shape of the observed object. However, changes in lights and radiation great affect their performance. Readings become noisier and unreliable, and consequently, a few parts of the surface become misread or even go missing. Even more so when the sensors deployed have lower specifications due to provisioning issues or cost.

The robot must interact with arbitrarily shaped objects. The lack of 3D models of these objects requires the use of additional sensors, *i.e.* 3D cameras, to perceive the position and shape of the object. In normal conditions, high-end off-the-shelf sensors capture highly accurate readings of the object, and the interpolation of the recorded points produces a precise 3D shape of the observed object. However, changes in lights and radiation great affect their performance. Readings become noisier and unreliable,

and consequently, a few parts of the surface become misread or even go missing. Even more so when the sensors deployed have lower specifications due to provisioning issues or cost.

Contributions

We show how to plan serial manipulator cutting paths on smooth but otherwise arbitrarily shaped surfaces, observed as a noisy Point Cloud (PC). We show how points from the PC can be used to generate a net of possible path nodes, by mapping them from Euclidean space onto a Riemannian manifold in which sampling-based path-planning takes place. We further show how a cost function can be constructed, which considers kinematics and reachability constraints while evaluating the manipulability of the robot throughout its motion along any candidate path.

The main contributions are:

- we propose a procedure to project observed points from the PC onto the Riemannian manifold of the object;
- we design a bi-objective cost function, which maximises manipulability while also reducing overall path length during path optimisation;
- we incorporate the cost function into an RRT* planner, we call the new approach RRT*-RMM (RRT* with Riemannian Manifold mapping and Manipulability cost);
- we empirically evaluate the planner with both real and simulated robots on various surface shapes.

3.2 Mathematical formulation

Given two points A and B (possibly given by the user), we want to generate a path for the robot end-effector that stays in touch to the object's surface throughout the entire

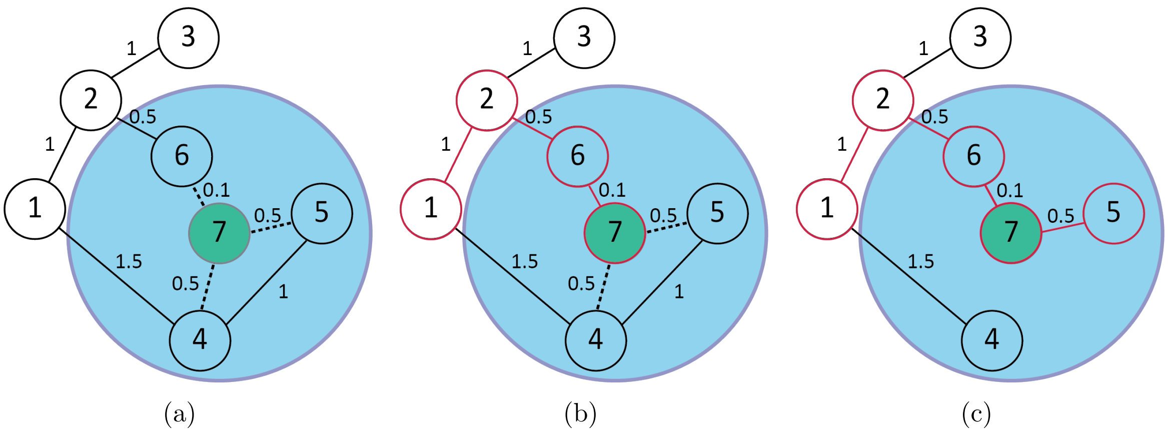


Figure 3.1: The picture displays a RRT* iteration. Fig. 3.1(a), a new point (in green) is evaluated. We compute the smaller cost to reach 7 from 1, Fig. 3.1(b). The algorithm checks nodes nearby 7 for rewiring. The previous route to reach 5 from 1 has a cost of 2.5 via the node 5. The rewiring finds a new route through the new node, 7, with smaller cost 2.1. The tree is updated accordingly, Fig. 3.1(c).

path and connects A with B. The PC representing the object is noisy, and we propose to apply the Riemannian geometry theory to overcome the uncertainties. We leverage the manipulability index to obtain a path with a greater likelihood of success. Hereafter, we introduce the main concepts we will use in the rest of our manuscript.

3.2.1 Rapidly-exploring Random Tree

Rapidly-exploring Random Tree (RRT) is one of the most common sampling-based path planners, [VZ11]. The basic idea behind RRT is to sample points within a space of interest, *e.g.* either Operational or configuration space, and add them in a tree structure based on a metric accounting for the distance. At every iteration, the algorithm generates a new point checks whether is collision-free or not and takes into account constraints imposed by, *e.g.*, the robot kinematics. Then, it connects that point to the closest node in the tree.

Fig. 3.1(a) shows an algorithm iteration, where the new point, 7, is evaluated. We compute the cost to reach 7 from all nodes within the blue circle (node: 4, 5, 6). The node whose cost is lesser becomes the parent of 7, Fig. 3.1(b). RRT* [KF11] is an extension to the classical RRT, which allows the re-evaluation of nodes already in the tree when a new point is available. This procedure is usually referred to as *rewiring*. During the rewiring, the algorithm selects the neighbourhood of a point (points in the tree within a range distance to the point) and evaluates whether these nodes improve their value passing through the new available point.

Fig. 3.1(c) shows the *rewiring* process for the node 7. The algorithm checks whether a new route for reaching 4, 5, and 6 is now available via 7. At first, the cost to reach 5 from 1 via 4 was 2.5. Thanks to the new node, a new route is available through 7 with a cost of 2.1. Hence, the tree is updated using this information, and node 7 becomes the parent of node 5. This process provides RRT with a better convergence rate, and the solution converges to the shortest path as the number of samples goes to infinity.

3.2.2 Robotic representation

Let $\boldsymbol{q} \in \mathbb{R}^n$ represent the robot configuration where n is the number of degrees of freedom (dof) of the robot. Given a specific $\boldsymbol{q}$, the position and orientation of every point of the robot are uniquely defined (forward kinematics). This mapping, $\boldsymbol{f_r}$, is commonly expressed as

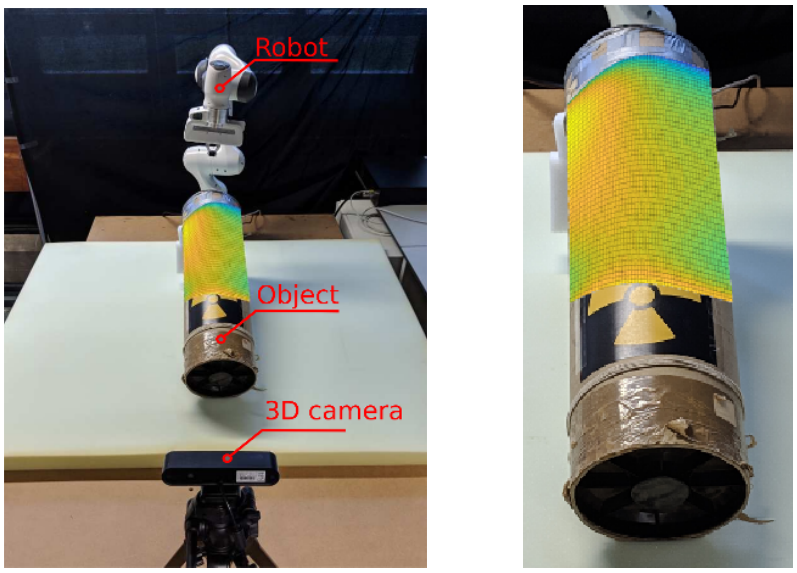
$$\boldsymbol{r} = \boldsymbol{f_r}(\boldsymbol{q}), \tag{3.1}$$

where $\boldsymbol{r} \in \mathbb{R}^m$ is the position and/or the orientation of a point of interest of the robot in the Cartesian space and m is the dimension of this representation (*e.g.*, $m = 3$ for 3D position, $\boldsymbol{r} = [\boldsymbol{p}]$; or $m = 6$ for 3D position ($\boldsymbol{p}$) and Euclidean orientation ($\boldsymbol{\psi}$), $\boldsymbol{r} = [\boldsymbol{p} \ \ \boldsymbol{\psi}]^T$)). Differential kinematics are defined using the robot Jacobian $\boldsymbol{J}(\boldsymbol{q})$ as

$$\dot{\boldsymbol{r}} = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{3.2}$$

and bridge the velocities in the configuration space to velocities in the Operational space[1].

[1]Since the Jacobian matrix always depends on the configuration $\boldsymbol{q}$, we drop the dependence on $\boldsymbol{q}$, and in the following we write $\boldsymbol{J}(\boldsymbol{q})$ as $\boldsymbol{J}$ when no confusion arises.



(a) (b)

Figure 3.2: An experimental robotic cutting setup (Fig. 3.2(a)). A 3-D camera (positioned in front of the robot at 2.8 [m] distance facing the robot) captures the point cloud of the object surface. Our approach computes a cutting path with given initial and end points. This path is suitable for the robot kinematics as our algorithm accounts for a manipulability index. Fig. 3.2(b) shows the manipulability of the robot: each point is coloured based on the manipulability corresponding to the configuration the robot is in when its end effector touches such point. Points with lower manipulability are shown in blue; points with higher manipulability are represented in yellow and red.

3.2.3 Manipulability

If we constrain the norm of the configuration velocities to be unitary, the configuration lies on the unitary sphere $\mathbb{S}^1$

$$|\dot{\boldsymbol{q}}| = \dot{\boldsymbol{q}}^T\dot{\boldsymbol{q}} = \dot{\boldsymbol{r}}^T\boldsymbol{J}^{\dagger T}\boldsymbol{J}^{\dagger}\dot{\boldsymbol{r}} = \dot{\boldsymbol{r}}^T\boldsymbol{\Gamma}^{\dagger}\dot{\boldsymbol{r}} = 1 \tag{3.3}$$

where † is the inverse matrix whether $\boldsymbol{J}$ is square or the pseudo-inverse matrix otherwise. Previous work leverages manipulability to yield optimal manipulation movements for planning a suitable grasping pose [GEAFRGS17]. We would also like to optimise the manipulation capability for robotic cutting. The conventional measure of manipulability [Yos85b] is defined as

$$w(\boldsymbol{q}) = \sqrt{det(\boldsymbol{\Gamma})} = \sqrt{\lambda_1\lambda_2...\lambda_n}, \tag{3.4}$$

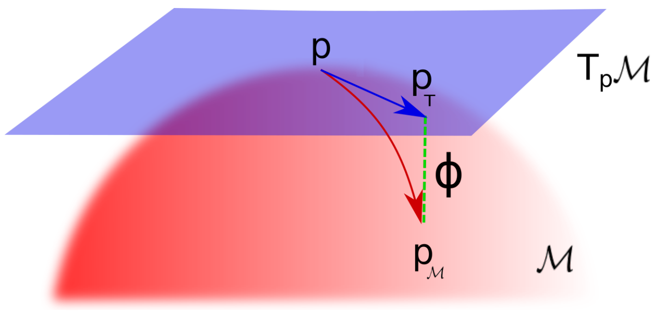


Figure 3.3: Given a smooth manifold $\mathcal{M}$, showed in red, the neighbourhood of every point $\boldsymbol{p}$ onto the manifold can be approximated with a tangent plane $T_p\mathcal{M}$, coloured in blue. The function ϕ uniquely maps every point $\boldsymbol{p}' \in T_p\mathcal{M}$ onto the manifold $\mathcal{M}$, namely $\boldsymbol{p}''$.

where λ_i are the eigenvalues of $\boldsymbol{\Gamma}$. This index provides a value that is proportional to the volume of the manipulability ellipsoid, and it does not require a long computational time.

3.2.4 Riemannian Manifold

A manifold is an n-topological space that approximates the Euclidean space in the neighbourhood of its points [LN14]. A smooth manifold is a differentiable manifold in which all the transition maps are smooth. That is, derivatives of all orders exist, so it is a C^k-manifold for all k.

A Riemannian manifold is a pair $[\mathcal{M}, z]$ where $\mathcal{M}$ is a smooth manifold and z is an inner product of two vector spaces on the manifold. The family of these inner products represents the Riemannian metric.

In Fig. 3.3, a smooth manifold $\mathcal{M}$ is shown using red colour, and in blue the figure shows the tangent plane to $\mathcal{M}$ at a point, $\boldsymbol{p} \in \mathcal{M}$, namely $T_p\mathcal{M}$.

By definition of Riemannian manifold, given a point $\boldsymbol{p}_T \in T_p\mathcal{M}$ and a manifold $\mathcal{M}$, as per Fig. 3.3, a function which maps the point $\boldsymbol{p}_T$ onto the manifold $\mathcal{M}$ exists [LN14]. Applying the mapping to the point $\boldsymbol{p}_T$, we obtain a point $\boldsymbol{p}_\mathcal{M} \in \mathcal{M}$, which is the projection of $\boldsymbol{p}_T$ onto $\mathcal{M}$. The red vector connecting $\boldsymbol{p}$ and $\boldsymbol{p}_\mathcal{M}$ in Fig. 3.3 is unique, and it represents the shortest travelling distance between the two points lying onto the manifold, namely the geodesic distance. The map between points onto the plane $T_p\mathcal{M}$ and points onto the manifold $\mathcal{M}$ is called *exponential* map, $\phi : T_p\mathcal{M} \mapsto \mathcal{M}$, and it is defined as per eq. (3.5).

$$\boldsymbol{p}_{\mathcal{M}} = \phi(\boldsymbol{p}_T) = \exp_p(\boldsymbol{p}_T) = cos(\boldsymbol{p}_T)\boldsymbol{p} + sin(\boldsymbol{p}_T) \tag{3.5}$$

Conversely, the inverse mapping that projects point belonging to the manifold, $\mathcal{M}$, onto the tangent plane, $T_p\mathcal{M}$, is called *logaritmic* map, $\phi^{-1} : \mathcal{M} \mapsto Tp_{\mathcal{M}}$.

$$\boldsymbol{p}_T = \phi^{-1}(\boldsymbol{p}_T) = \log_p(\boldsymbol{p}_{\mathcal{M}}) \tag{3.6}$$

3.3 RRT*-RMM

Given a Point Cloud of an object, which includes a set of points, *i.e.*, $\mathbb{P} = \{\boldsymbol{p}^1_{\mathbb{P}}, ..., \boldsymbol{p}^m_{\mathbb{P}}\}$, where $\boldsymbol{p}_{\mathbb{P}} \in \mathbb{R}^3$ and m is the number of points in the set, we need to compute a series of points representing a cutting path between a given initial point $\boldsymbol{p}^A_{\mathbb{P}} \in \mathbb{P}$, and a given terminal point $\boldsymbol{p}^B_{\mathbb{P}} \in \mathbb{P}$. We use standard inverse kinematics to obtain the joint space configuration, $\boldsymbol{q}^A$, corresponding to $\boldsymbol{p}^A_{\mathbb{P}}$. We assume a PC captured by a depth sensor is a noisy data of a smoothly curved object, *i.e.*, $\boldsymbol{p}_{\mathbb{P}} = \hat{\boldsymbol{p}}_{\mathcal{M}} + \sigma$, where $\sigma \in \mathbb{R}^3$ is white noise and $\hat{\boldsymbol{p}}_{\mathcal{M}}$ is a point sampled from the manifold. We assume the set of points $\hat{\boldsymbol{p}}_{\mathcal{M}}$ to represent (discretely) a smooth (Riemannian) manifold. We aim to build a tree structure whose nodes represent a path suitable for a cutting task. Every node of the tree η includes a robot's end-effector (EE) pose suitable for cutting, $\boldsymbol{r} = [\boldsymbol{p}_{\mathcal{M}},\ \boldsymbol{\psi}]^T$ where $\boldsymbol{p}_{\mathcal{M}}$ and $\boldsymbol{\psi}$ denote the corresponding position and orientation, and the corresponding robot configuration $\boldsymbol{q}$, *i.e.* $\eta = \{\boldsymbol{p}_{\mathcal{M}}, \boldsymbol{\psi}, \boldsymbol{q}\}$. The desired tree includes a series of points $\boldsymbol{\xi} = \{\boldsymbol{p}^1_{\mathcal{M}}, \boldsymbol{p}^2_{\mathcal{M}}, ..., \boldsymbol{p}^n_{\mathcal{M}}\}$, where $\boldsymbol{p}^i_{\mathcal{M}} \in \mathcal{M}$, and $\boldsymbol{\xi}$ connects the desired initial, $\boldsymbol{p}^1_{\mathcal{M}} = \boldsymbol{p}^A_{\mathbb{P}}$, and terminal points, $\boldsymbol{p}^n_{\mathcal{M}} = \boldsymbol{p}^B_{\mathbb{P}}$. We assume the orientation of the end-effector is determined by a cutting task, *e.g.*, for cutting with a knife, the EE must be normal to $\mathcal{M}$ and the the knife needs to cut the object moving from $\boldsymbol{p}^{i-1}_{\mathcal{M}}$ to $\boldsymbol{p}^i_{\mathcal{M}}$.

At the *j-th* iteration of our algorithm, a point from the PC is randomly selected,

$\boldsymbol{p}_{\mathbb{P}}^{rand} \in \mathbb{P}$. Then, the algorithm finds a tree's vertex, $\eta^j = \{\boldsymbol{p}_{\mathcal{M}}^{nearest}, \boldsymbol{\psi}, \boldsymbol{q}\}$, whose point, $\boldsymbol{p}_{\mathcal{M}}^{nearest}$, has the minimum distance to $\boldsymbol{p}_{\mathbb{P}}^{rand}$. Because we do not have a mathematical expression of the object surface, which can be represented by $\mathcal{M}$, we cannot compute a plane tangent to $\mathcal{M}$. However, we can approximate the tangent plane at point $\boldsymbol{p}_{\mathcal{M}}^{nearest}$, namely $T_p\mathcal{M}$, by applying principal component analysis (PCA) on the points in the close neighbourhood of $\boldsymbol{p}_{\mathcal{M}}^{nearest}$. We now project the point $\boldsymbol{p}_{\mathbb{P}}^{rand}$ onto $T_p\mathcal{M}$, we name the projected point as $\boldsymbol{p}_T \in T_p\mathcal{M}$. Then, we compute the point $\boldsymbol{p}_T^{\beta}$ using the projected and nearest points as per eq. 3.7 by choosing a very small value for the step size β. This ensures the updated value on the tangent plane will be correctly mapped onto the Riemannian manifold through the exponential mapping in the next steps of the algorithm.

$$\boldsymbol{p}_T^{\beta} = \boldsymbol{p}_{\mathcal{M}}^{nearest} + \beta(\boldsymbol{p}_T - \boldsymbol{p}_{\mathcal{M}}^{nearest}). \tag{3.7}$$

Fig. 3.4 summarises the steps for projecting points from the PC to the manifold. This figure shows a sample PC, the underlying smooth manifold in red, and the tangent plane to $\boldsymbol{p}_{\mathcal{M}}^{nearest}$, $T_p\mathcal{M}$, is shown in blue colour. In Fig. 3.4(b), $\boldsymbol{p}_T$ is the projection of $\boldsymbol{p}_{\mathbb{P}}^{rand}$ onto $T_p\mathcal{M}$. We choose β in eq. 3.7 to be a positive definite value with $\beta \ll 1$, this assures us that the projected point on the tangent plane, $\boldsymbol{p}_T^{\beta}$, is very close to $\boldsymbol{p}_{\mathcal{M}}^{nearest}$, Fig. 3.4 (c), which satisfies the underlying assumption for exponential mapping from tangent plane to the Riemannian manifold, $\mathcal{M}$, [LN14]. Note that $\boldsymbol{p}_{\mathcal{M}}^{nearest}$, $\boldsymbol{p}_T^{\beta}$, and $\boldsymbol{p}_T$ lie on the tangent plane, Fig. 3.4 (c). Finally, we obtain a new point, $\boldsymbol{p}_{\mathcal{M}}^{new} \in \mathcal{M}$ using exponential mapping, as per eq. (3.5). In order to account for the kinematic of the manipulator in our cutting path planning, we introduce a modified cost to be used in our RRT* algorithm. We use (1) a cost accounting for the distance, denoted by C_d, and (2) a cost for manipulability, denoted by $C_{\boldsymbol{M}}$.

As for C_d, we compute the sum of all segments over the path to reach $\boldsymbol{p}_{\mathcal{M}}^{new}$ starting from the root of the tree, $\boldsymbol{p}_{\mathcal{M}}^{1}$. That is the sum of Euclidean distances between consecutive points in the tree $\{\bar{\boldsymbol{p}}_{\mathcal{M}}^{1}, \bar{\boldsymbol{p}}_{\mathcal{M}}^{2}, ..., \bar{\boldsymbol{p}}_{\mathcal{M}}^{I}\}$, which is the sequence of points from the root of the

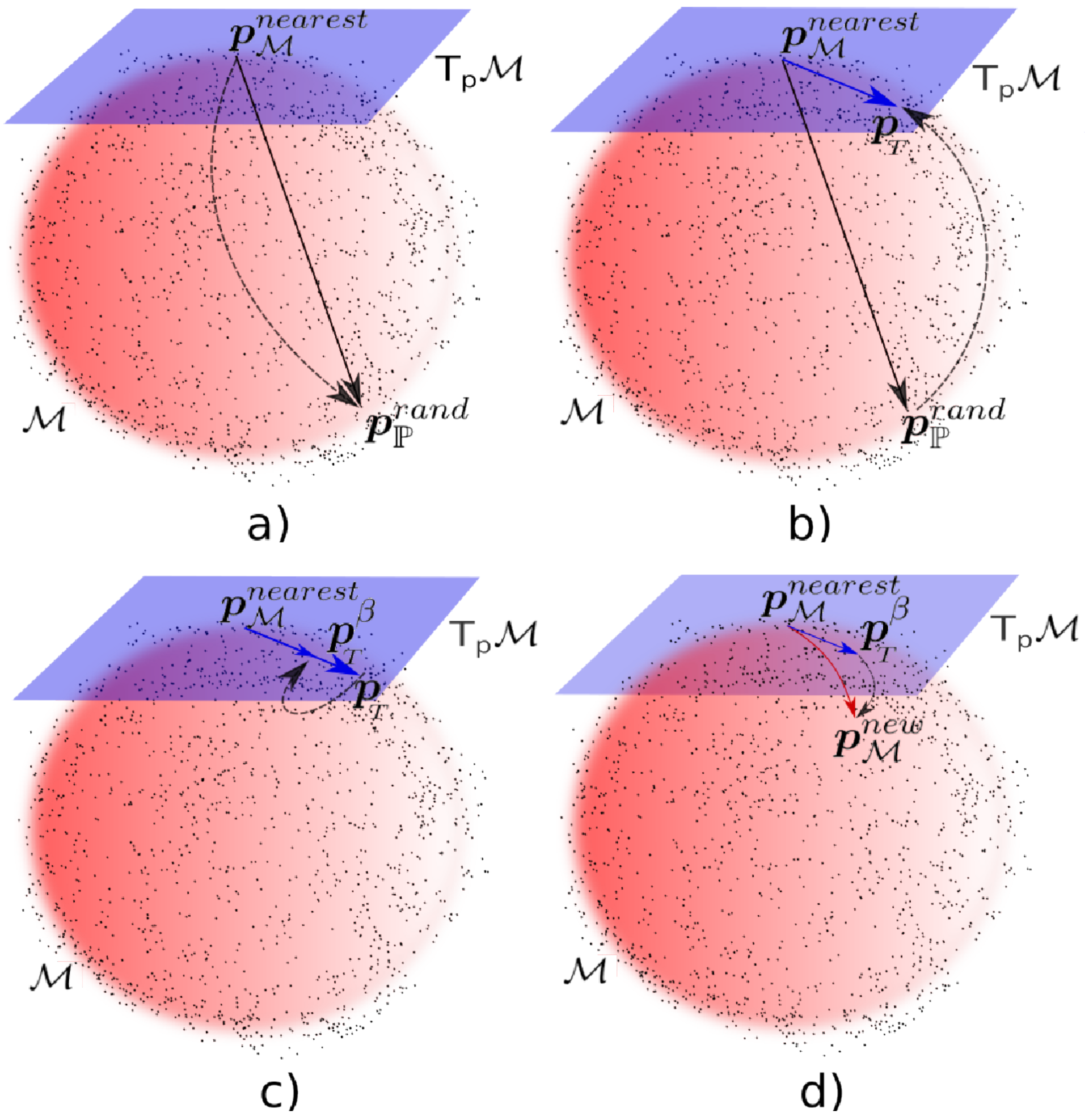

$\boldsymbol{p}_{\mathcal{M}}^{nearest}$
$\mathrm{T_p}\mathcal{M}$
$\mathcal{M}$
$\boldsymbol{p}_{\mathbb{P}}^{rand}$
a)
$\boldsymbol{p}_{\mathcal{M}}^{nearest}$
$\mathrm{T_p}\mathcal{M}$
$\boldsymbol{p}_{T}$
$\mathcal{M}$
$\boldsymbol{p}_{\mathbb{P}}^{rand}$
b)
$\boldsymbol{p}_{\mathcal{M}}^{nearest}$
$\boldsymbol{p}_{T}^{\beta}$
$\boldsymbol{p}_{T}$
$\mathrm{T_p}\mathcal{M}$
$\mathcal{M}$
c)
$\boldsymbol{p}_{\mathcal{M}}^{nearest}$
$\boldsymbol{p}_{T}^{\beta}$
$\mathrm{T_p}\mathcal{M}$
$\boldsymbol{p}_{\mathcal{M}}^{new}$
$\mathcal{M}$
d)

Figure 3.4: This image shows the procedure to obtain a new point on the manifold from a point on the Point Cloud (PC). We sample a point from the PC, $\mathbb{P}$, and obtain $\boldsymbol{p}_{\mathbb{P}}^{rand}$, Fig.3.4(a). We project $\boldsymbol{p}_{\mathbb{P}}^{rand}$ onto the tangent manifold $T_p\mathcal{M}$ and obtain $\boldsymbol{p}_T$, Fig.3.4(b). Then, we take a step starting from $\boldsymbol{p}_{\mathcal{M}}^{nearest}$ towards $\boldsymbol{p}_T$ using β, and obtain $\boldsymbol{p}_T^{\beta}$, Fig.3.4(c). Finally, Fig.3.4(d) shows the mapping of the point $\boldsymbol{p}_T^{\beta} \in T_p\mathcal{M}$ to the manifold $\mathcal{M}$ using the map ϕ, thus obtaining $\boldsymbol{p}_{\mathcal{M}}^{new}$.

tree to the point $\boldsymbol{p}_{\mathcal{M}}^{new}$, as per eq. (3.8).

$$C_d(\boldsymbol{p}_{\mathcal{M}}^{new}) = g(\bar{\boldsymbol{p}}_{\mathcal{M}}^{I}, \boldsymbol{p}_{\mathcal{M}}^{new}) + \sum_{i=1}^{I} g(\bar{\boldsymbol{p}}_{\mathcal{M}}^{i}, \bar{\boldsymbol{p}}_{\mathcal{M}}^{i-1}) \tag{3.8}$$

where I is the number of points visited in the tree before reaching $\boldsymbol{p}_{\mathcal{M}}^{new}$ and $g(.)$ is the geodesic distance between two adjacent points in the path. Assuming adjacent points are very close, this geodesic distance can be approximated by $g(\boldsymbol{p}^i, \boldsymbol{p}^{i-1}) = \|\boldsymbol{p}^i - \boldsymbol{p}^{i-1}\|$.

The manipulability cost is also computed over the path, as per eq. (3.9).

$$C_{\boldsymbol{M}}(\boldsymbol{q}^{new}) = \frac{1}{I+1}\left(\frac{1}{w(\boldsymbol{q}^{new})} + \sum_{i=1}^{I}\frac{1}{w(\boldsymbol{q}^i)}\right) \tag{3.9}$$

where $\boldsymbol{q}^{new} = \boldsymbol{q}^{nearest} + \boldsymbol{J}^{\dagger}(\boldsymbol{r}^{new} - \boldsymbol{r}^{nearest})$ is the corresponding configuration computed using the differential kinematics to match the displacement between $\boldsymbol{p}_{\mathcal{M}}^{nearest}$ and $\boldsymbol{p}_{\mathcal{M}}^{new}$, $\boldsymbol{q}^1 = \boldsymbol{q}^A$, and w is the manipulability index presented in eq. (3.4). We use the inverse of w so that minimising the manipulability cost is equivalent to maximising manipulability.

Eventually the cost is a weighted sum of $C_{\boldsymbol{M}}$ and C_d, as per eq. (3.10).

$$C(\boldsymbol{p}_{\mathcal{M}}^{new}, \boldsymbol{q}^{new}) = (1-\alpha)C_d(\boldsymbol{p}_{\mathcal{M}}^{new}) + \alpha C_{\boldsymbol{M}}(\boldsymbol{q}^{new}) \tag{3.10}$$

The coefficient $\alpha \in [0, 1]$ sets a trade-off between $C_{\boldsymbol{M}}$ and C_d. The proposed algorithm turns into the classical RRT* when $\alpha \to 0$; while $\alpha \to 1$ means only the manipulability is considered for planning the cutting path. This parameter must be chosen based on the corresponding domain knowledge, *e.g.*, we may need a different value of α for cutting with a knife or for cutting with a rotatory tool.

3.4 RRT*-RMM outline

RRT*-Riemannian Mapping Manipulability (RRT*-RMM) iteratively builds a tree of nodes. Every node is composed of the pose $\boldsymbol{r}$ (position and orientation) and the relative configuration of the robot.

The algorithm takes as inputs: the initial and terminal position of the end-effector, the pointcloud of the object to cut, and the kinematic of the robot. The KD-tree structure is initialised using the initial pose and configuration as root.

The algorithm allows two iterative methods: it stops when a feasible path is found or runs for a fixed number of iterations. The former approach permits obtaining paths with

Algorithm 1: RRT*-Riemannian-mapping-manipulability

```
param: $\eta = \{\boldsymbol{r},\ \boldsymbol{q}\}$, is a tree node;
       $\boldsymbol{r} = [\boldsymbol{p}_{\mathcal{M}},\ \boldsymbol{\psi}]$ is end-effector pose on $\mathcal{M}$
1  tree.init($\eta^1$);
2  while $n \leq N$ do
3      $\boldsymbol{p}_{\mathbb{P}}^{rand} \leftarrow getPointFromPntCloud()$;
4      $\boldsymbol{r}^{nearest},\ \boldsymbol{q}^{nearest} \leftarrow Nearest(\boldsymbol{p}_{\mathbb{P}}^{rand})$;
5      $T_p\mathcal{M} \leftarrow computeTangentPlane(\boldsymbol{p}_{\mathcal{M}}^{nearest})$;
6      $\boldsymbol{p}_T \leftarrow projectOnTangentPlane(\boldsymbol{p}_{\mathbb{P}}^{rand}, T_p\mathcal{M})$;
7      $\boldsymbol{p}_T^{\beta} \leftarrow takeAStep(\boldsymbol{p}_T, \boldsymbol{p}_{\mathcal{M}}^{nearest})$;
8      $\boldsymbol{p}_{\mathcal{M}}^{new} \leftarrow expRiemannianMap(\boldsymbol{p}_T^{\beta})$;
9      $\boldsymbol{\psi}^{new} \leftarrow getTangentPlaneNormal(T_p\mathcal{M})$;
10     $\boldsymbol{r}^{new} \leftarrow [\boldsymbol{p}_{\mathcal{M}}^{new},\ \boldsymbol{\psi}^{new}]^T$;
11     $\boldsymbol{q}^{new} \leftarrow \boldsymbol{q}^{nearest} + J^{\dagger}(\boldsymbol{q}^{nearest})(\boldsymbol{r}^{new} - \boldsymbol{r}^{nearest})$;
12     if $ObstacleFree(\boldsymbol{q}^{new})$ then
13         $\boldsymbol{R}^{near} \leftarrow getSetofNearVertices(\boldsymbol{r}^{new})$;
14         for $h = 1$ to $H = |\boldsymbol{R}^{near}|$ do
15             $C_d \leftarrow computeDistance(\boldsymbol{r}^{new}, \boldsymbol{R}^{near}(h))$;
16             $C_{\boldsymbol{M}} \leftarrow computeMan(\boldsymbol{r}^{new}, \boldsymbol{R}^{near}(h))$;
17             $\boldsymbol{C}(h) = (1-\alpha)C_d + \alpha C_{\boldsymbol{M}}$;
18         end
19         $\boldsymbol{r}^{best} \leftarrow selectMinimumElement(\boldsymbol{C})$;
20         tree.addVertex($\eta^{new}$);
21         tree.addEdge($\eta^{best}$, $\eta^{new}$);
22         tree.rewire($\boldsymbol{R}^{near}$, $\eta^{new}$);
23     end
24     discard the point;
25 end
26 return tree;
```

lesser length and higher manipulability.

From *line 4*, the main loop starts. The first step samples a point from the Point Cloud, *line 5.* Then, a query to the tree searches for the nearest node to the picked point obtaining the $\eta = [\boldsymbol{p}_{\mathcal{M}}^{nearest},\ \boldsymbol{\psi}]$. The position of the nearest node is used by the algorithm to compute the tangent plane onto the surface, $T_p\mathcal{M}$ in *line 7.* To attain points on the real surface, we leverage the Riemannian geometry. First of all, we project the sampled point to the tangent plane. Then, we take a step toward the projected point. The small size of the step allows the approximation assumption to stand throughout the iterations. Finally, we use the exponential map to obtain the point on the object's manifold and the orientation via the tangent plane normal, *lines 8-11*.

Line 13 computes the new configuration using the differential kinematics, and *line 14* checks whether the robot is in collision with any obstacle or not.

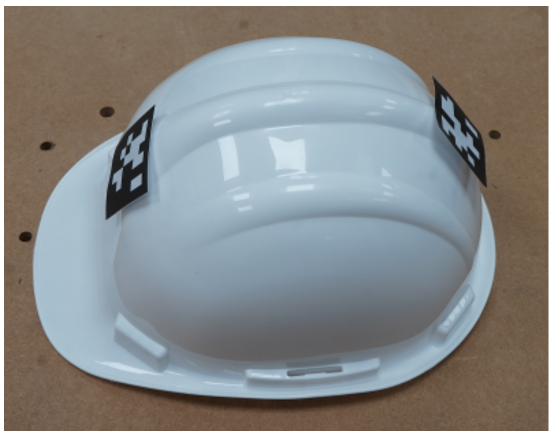
From *line 15* to *20*, the algorithm computes the cost for the new node and finds the best parent in the tree. The parent node is elected from the near nodes already in the tree. The for loop at *line 14* loops on the cardinality of R^{near}. Finally, the tree connects the new node. The last step of the loop launches the rewiring of the tree. This function re-evaluates vertices in the tree and checks if every node improves its cost passing through the new vertex.

The algorithm returns the tree structure, which can be efficiently searched for obtaining the path to the goal.

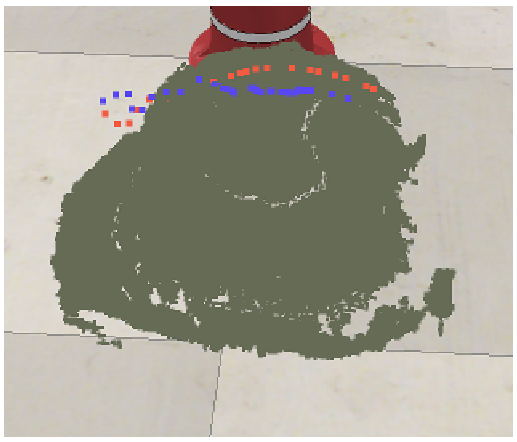
3.5 Experimental validation of RRT*-RMM

We use a Panda robot manufactured by Franka EMIKA for the real-world experiments We also provide some results with Sawyer in V-REP[1]. Although both Panda and Sawyer are 7-DOF robotic arms equipped with a standard parallel jaw gripper, they have different kinematic chains. This shows how the proposed approach is readily transferable across different manipulators' kinematics. An Orbbec Astra RGB-D camera scans the area in front of the robot (Fig. 3.2), and we remove points outside the robot's workspace as preprocessing filtering on the point cloud. The camera is calibrated with respect to the robot base frame. As such, we can express the PC captured by the camera in the robot base frame. Furthermore, we attach two markers to each object, as shown in Fig. 3.5(a). These markers represent the start point A and the end point B of the path and allow a human operator to select the initial and end points of the cut. The RGB-D camera takes the point cloud of the scene in front of the robot as input, and our algorithm computes a

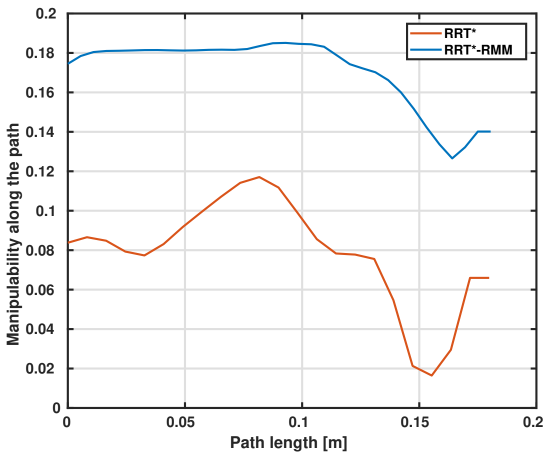
[1]Although the algorithm needs the robot kinematics, our ROS implementation takes the robot URDF directly from the ROS server parameter. Therefore, we present some data collected with Panda and some others with Sawyer to show the robustness of our approach to changes in the kinematic chains.



(a)



(b)



(c)

Figure 3.5: This image shows the path proposed by the two algorithms for a path connecting the two points indicated with the markers on a helmet in 3.5(a). The helmet has 3 ridges on its shell. The top part of the helmet consists of smooth surfaces with large curvatures, whereas the shell and the ridges are connected with very small curvature. 3.5(b) shows the point cloud of the helmet captured by the camera in V-REP and the path proposed by RRT* and our proposed approach are shown in red and blue dotted lines, respectively. In 3.5(c), the manipulability of both paths is shown.

cutting path between point A and point B.

Fig. 3.2 shows the experimental setup with a cylindrical container emulating a nuclear waste barrel. A heat map overlaid on the object represents the manipulability corresponding to the configuration the robot is in when its EE is at the point on the object. We use the standard inverse kinematics (IK) of the robot to compute the joint configurations. We developed a full ROS package to compute the optimal cutting path. As the robot URDF

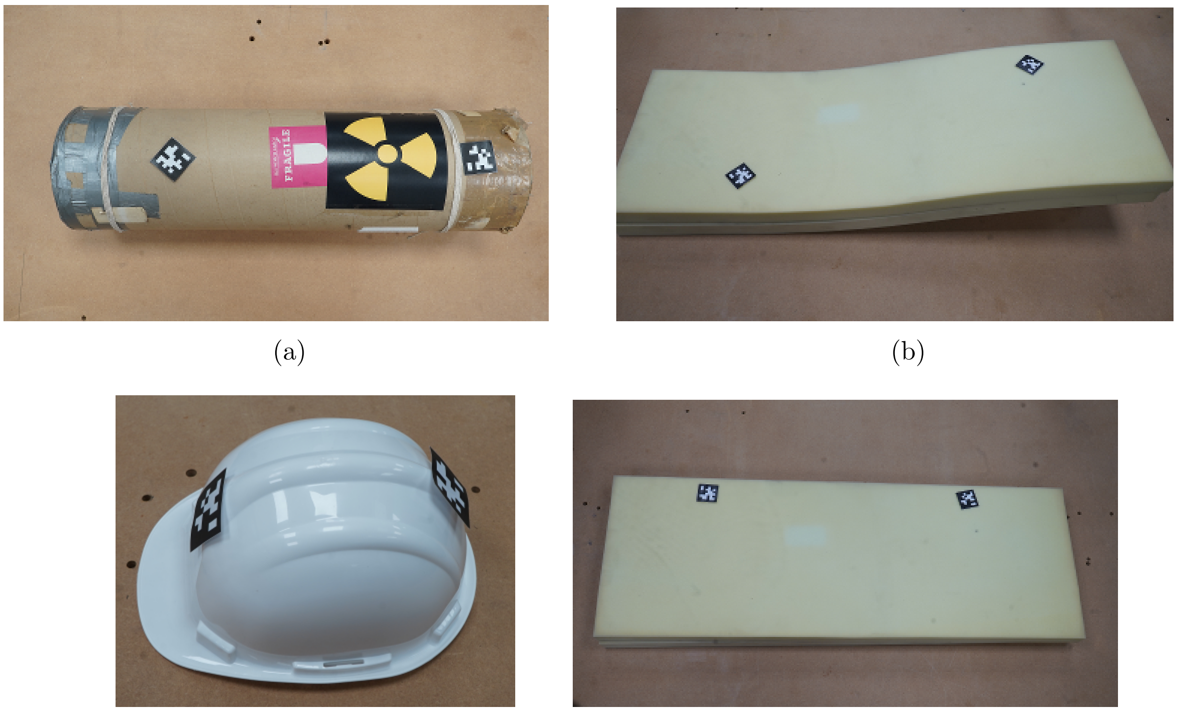


Figure 3.6: Test objects used for testing our algorithm, Fig. 3.6(a) a barrel, Fig. 3.6(b) a curved object, Fig. 3.6(c) a helmet and Fig. 3.6(d) a flat object. The pictures also show the position of the markers.

can be loaded onto the parameter server and used by the algorithm, we can easily repeat our computation with any manipulator whose URDF is available.

We used four objects (Fig. 3.6) to illustrate the effectiveness of our approach in generating a cutting path on different objects' surfaces. These objects are a barrel (cylindrical container), a curved object (made of foam), a safety helmet and a flat object (also made of foam). These objects represent typical objects that are to be cut in a nuclear environment.

Fig. 3.5 shows the helmet we used for our experiments along with the markers attached to the object. The markers fix the initial and end point of the cutting path. These points can be provided by a human operator during real-world deployments. Fig. 3.5(b) shows the point cloud of the helmet captured by the camera and visualised in V-REP. The paths computed by RRT* and our proposed approach, RRT*-RMM, are shown with red and blue dotted lines, Fig. 3.5(b). These results show that RRT* and RRT*-RMM effectively generate cutting paths on the object surface. Fig. 3.5(c) also shows the manipulability

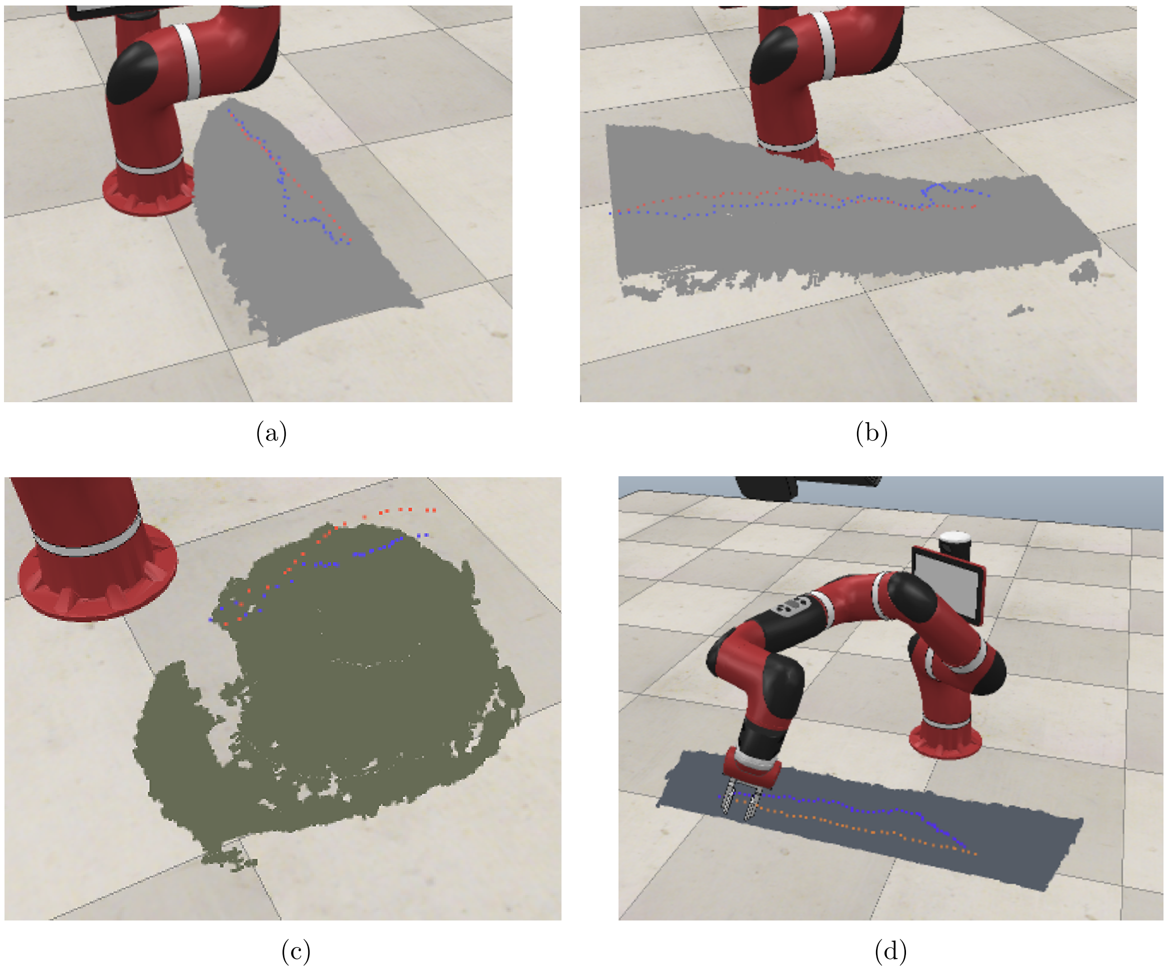


Figure 3.7: Paths proposed by RRT* and RRT*-RMM for the four objects, using initial and target positions shown in Fig. 3.6 using the markers. Fig. 3.7(d) shows how RRT* proposes a path close a singularity for the robot (orange path) instead, the RRT*-RMM does a semicircle route to avoid it. In our experiments, we empirically selected $\alpha = 0.7$ to trade off path's length and manipulability.

corresponding to the paths obtained by RRT* and RRT*-RMM with red and blue lines, respectively. This figure shows that our algorithm finds a path that has significantly improved manipulability for the robot throughout the whole path.

We see that RRT* generates a path specific just to the shape of the object. In contrast, RRT*-RMM computes paths not only specific to the shape of the object but also specific to (i) the position of the object relative to the robot base frame and (ii) the kinematic chain of the robot. If we change either object position or the robotic arm, RRT*-RMM computes the new best path for that scenario. As such, our algorithm always finds a path that best fits the specific problem setting.

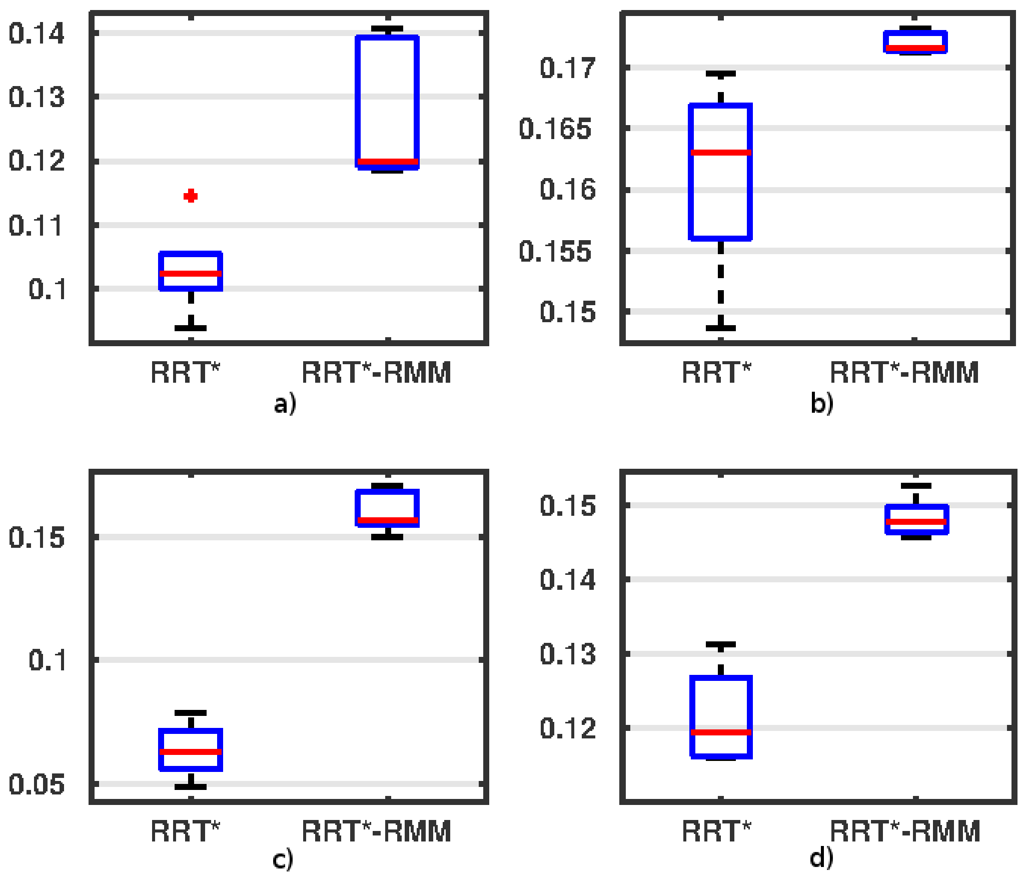


Figure 3.8: Box plot of the manipulability values obtained by RRT* and RRT*-RMM for all four objects. The order of these figures corresponds to the order of object figures shown in Fig. 3.6, a) the barrel, (b) the curved object, and (d) the flat object.

We performed similar experiments with all four objects shown in Fig. 3.6. Sample PCs of 4 objects visualised in V-REP are shown in Fig.3.7 along with the computed path by RRT* and RRT*-RMM, red and blue respectively. In detail, Fig. 3.7(d) shows that the robot faces singularity if it follows the path obtained by RRT* for the flat object shown in Fig. 3.6. In contrast, it does not experience this issue when using the path obtained with RRT*-RMM because the approach is explicitly designed to avoid such an issue. The paths visualised in V-REP in Fig. 3.7 correspond to the positions of the markers shown in Fig. 3.6. These figures show that slight differences in the path obtained by RRT* and RRT*-RMM yield higher manipulation capability for the manipulator.

Because the core of our planning algorithm is random, we need statistical evidence

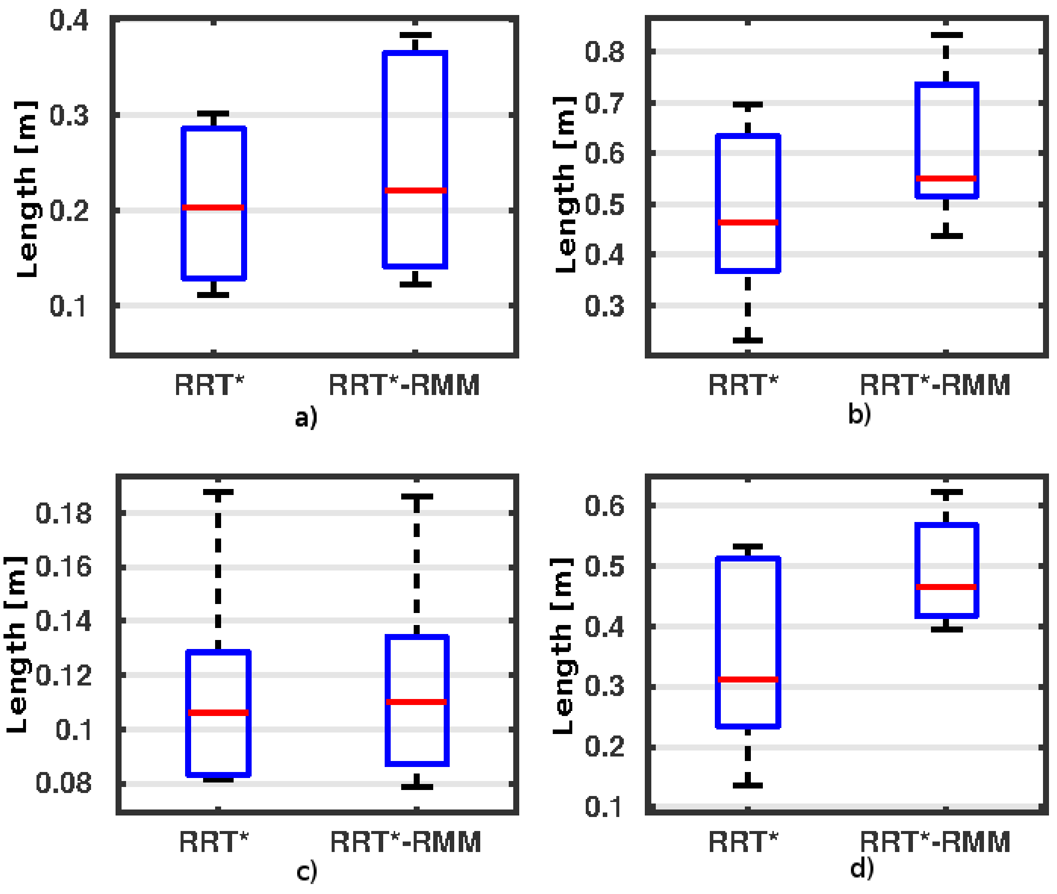


Figure 3.9: This figure shows the path length found for the same objects. Every object has been tested using five goal positions, and the plots include the data for all the trials. The order of these figures corresponds to the order of object figures shown in Fig. 3.6, a) the barrel, (b) the curved object, and (d) the flat object.

that our approach achieve the expected improved results. Therefore, we repeated the experiments five times. At every time, we change the end point for every object and generate the cutting paths using RRT* and RRT*-RMM. We collected the data on manipulability and length of the computed paths by RRT* and RRT*-RMM, and the corresponding box plots are shown in Figs. 3.8 and 3.9. The object surfaces we used are largely dissimilar, *e.g.* we have objects with different geometry, such as a flat surface and a helmet with sphere-like geometry. Fig. 3.8 shows the box plots of obtained manipulability for objects shown in Fig. 3.7. The mean and variance of the data summarised in the box plots in Fig. 3.8 suggest that RRT*-RMM yields paths with generally higher manipulability. Although the path computed by our algorithm and RRT* differs at each repetition due to (1) the randomness of node generation and (2) the chosen goal point, Fig. 3.8 shows our approach yields bigger manipulability values in a mean-wise. RRT*-RMM also yields a smaller variance of manipulability which is another desired characteristic of our algorithm.

As we expected, the path length is increased with respect to the RRT* baseline since it is leveraged for a higher manipulability throughout the path. Nonetheless, this increment is not very high as it has shown in Fig. 3.9. The increased path length is $\sim$10% for the barrel, the curved object and the safety helmet, and $\sim$50% for the flat object.

Fig. 3.10 reports the manipulability values yielded by our algorithm on three scenarios using the Panda robot: a plane surface, a curved surface and a cylindrical surface. The time is normalised between 0 and 1, where 0 is the beginning of the path and 1 is the end. We use the RRT* algorithm as a baseline to show the effectiveness of our approach. RRT*-RMM shows a greater manipulability value in all three cases.

In particular, the plane surface experiment passes through a singular point if we choose the shortest path, which yields a manipulability value of 0.001 (lower value of the blue line). However, RRT*-RMM avoids the singular configuration and reaches a far greater manipulability profile.

The curved surface experiment shows the same behaviour as the first. Our algorithm obtains higher values throughout the path. Moving up the hill allows the robot configurations with higher manipulability.

As for the last experiment, our approach greatly improves the manipulability within the time-span $[0, 0.65]s$ but obtains less manipulability in the last $0.35s$. Although RRT*-RMM targets the maximisation of manipulability, it further accounts for both object's surface and the robot's kinematic. This trade-off results in a path with better manipulability overall but a slightly less value in the last part.

We care to notice that the choice of the parameter α in eq. 3.10 allows us to weigh the manipulability component of the algorithm based on the cutting task we are performing,

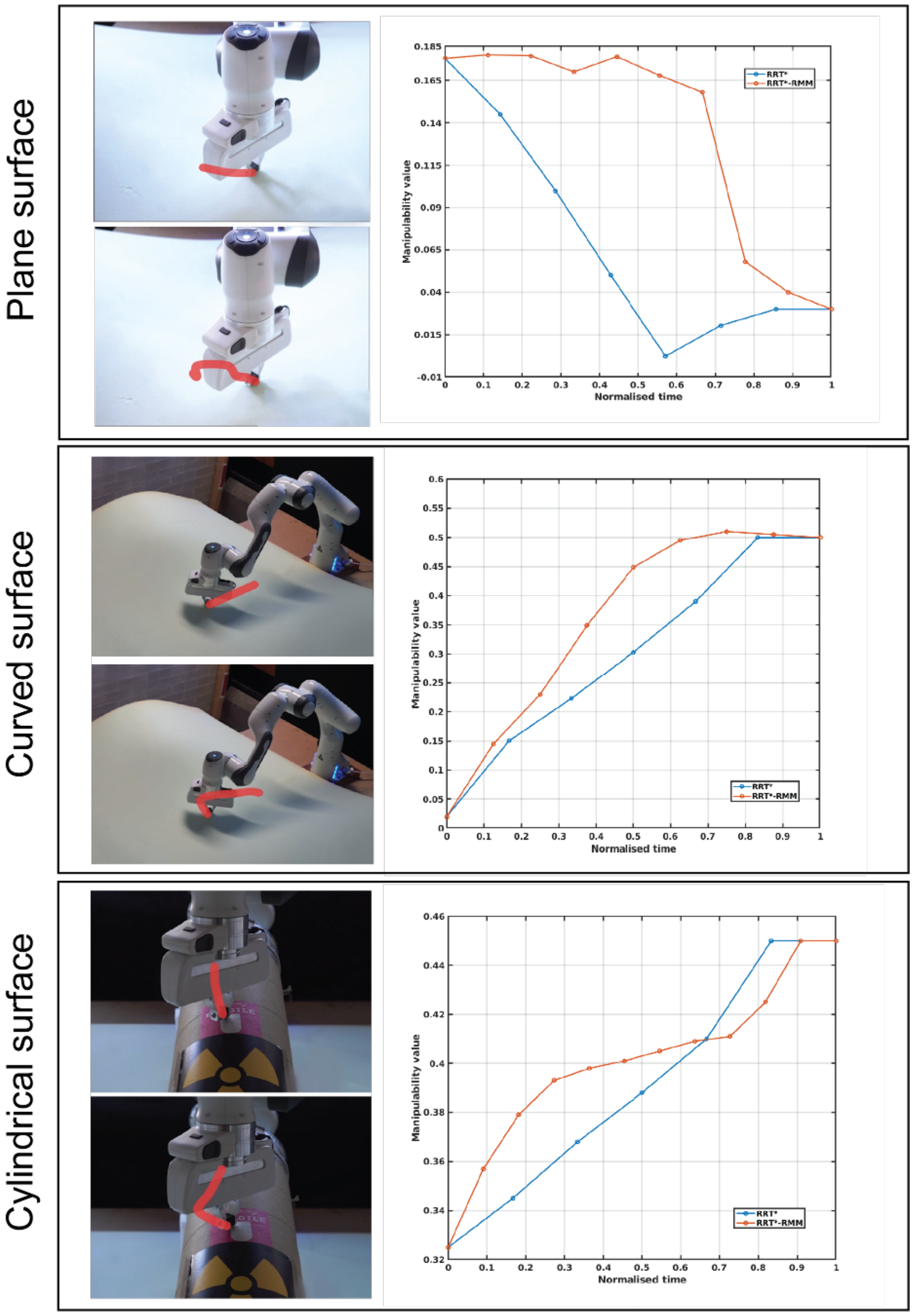


Figure 3.10: We test out our path planner on three real scenarios: a plane surface, a curved surface, and a cylindrical surface. We use the naive RRT* algorithm as a baseline for our approach. The RRT* generates a straight line between the start and destination point. However, for plane surface the robot passes through a singularity when the shortest path is selected. In all experiments, our method yields higher manipulability for reaching the destination.

e.g. cutting with a knife and rotating tool may need different manipulation capabilities. Although a value of α close to 1 might seems a reasonable option, it has the shortcoming of resulting in non-desirable long and very jerky cutting paths. In our experience, values below 0.85 lead to cutting paths with an acceptable length and increased manipulability along the path.

3.6 Conclusion

This chapter addresses the problem of constrained motion planning from vision, which enables a robot to move its end-effector on an observed surface, given start and destination points. We find robot trajectories which maximise the robot's manipulability throughout the motion. This work has application in industrial problems of robotic rough cutting. Our approach uses a mapping between Euclidean space and the Riemannian manifold to project the random samples taken from the Point Cloud onto the object's surface. This mapping step in our algorithm allows us to compute the path on an object surface using only a Point Cloud, without the need for a complete 3-D model. Moreover, we use a cost comprised of the sum of manipulability indices along the path and the distance travelled between the initial and evaluated points. The manipulability index added to the RRT* cost assures generated paths yielding higher manipulability values.

Here, we did not address the lower-level dynamics and control issues of the forceful contact between the robot and cut material while a cutting path is being followed. In contrast, we focused on the higher-level problem of planning a safe cutting path for the manipulator on a surface obtained by a vision system. Thereby, in our experiments, we inflated the Point Cloud, *i.e.*, we modified PC to surround the object surface with 0.02 [m] offset. Therefore, our algorithm computes a path safe for our real robot demonstrations, *i.e.*, the robot could move along the obtained path while keeping a 0.02 [m] standof from the object's surface to avoid forceful interaction with the object.

For proof of concept, here we demonstrate how to incorporate our cost function into the RRT* framework. However, our cost function may be adopted by any already existing numerical optimiser, and it is readily extendable to other common path planners, *e.g.*, RRT, PRM. We showed how our approach successfully yield the cutting paths using two different 7 dof arms. Nonetheless, no limitations on the manipulator dof has been considered.

Moreover, we considered only the surfaces with different shapes, which are smooth (Riemannian) manifolds. Our approach does not tackle neither surfaces with sharp discontinuities nor non-smooth surfaces. However, a possible solution is to consider the surface as a set of locally Riemannian patches and run the algorithm on every patch.The final cutting path will result as the union of the solutions found to the sub-problems.

We presented a series of experiments with a Panda robot and a Sawyer robot. The experiments include computing the cutting paths on four different objects. Since the core of RRT* and RRT*-RMM is random sample generation, we performed a statistical study that shows how RRT*-RMM improves the manipulability index while trading off the length of the cutting path, *i.e.*, RRT*-RMM obtains an increased manipulation capability (avoiding robot-related issues) at the cost of increased path length.

CHAPTER 4

Path planning for mobile manipulators subject to non-holonomic and task constraints

4.1 Introduction

The previous chapter addresses the planning of manipulability-aware paths for fixed-based arms. However, many scenarios require the robot to reach areas outside the workspace of the robot. Operations such as cutting a pipe might require the robot to extend either too far or too close from the base, with no configuration left feasible for the robot. Also, the object might have a complex shape, and the robot needs to go around the surface to complete the cut. Those further motions may bring the arm to cover long journeys that cross the boundaries of the workspace.

An alternative to fixed-based manipulators is floating-based robots. Mobile manipulators offer a wider range of motions and virtually infinite workspace. Generally, heavy-duty or rugged operations demand sturdy mobile platforms with caterpillar tracks or skid-steer rough-terrain wheels, Fig. 4.1. This category of robots is called *non-holonomic* mobile

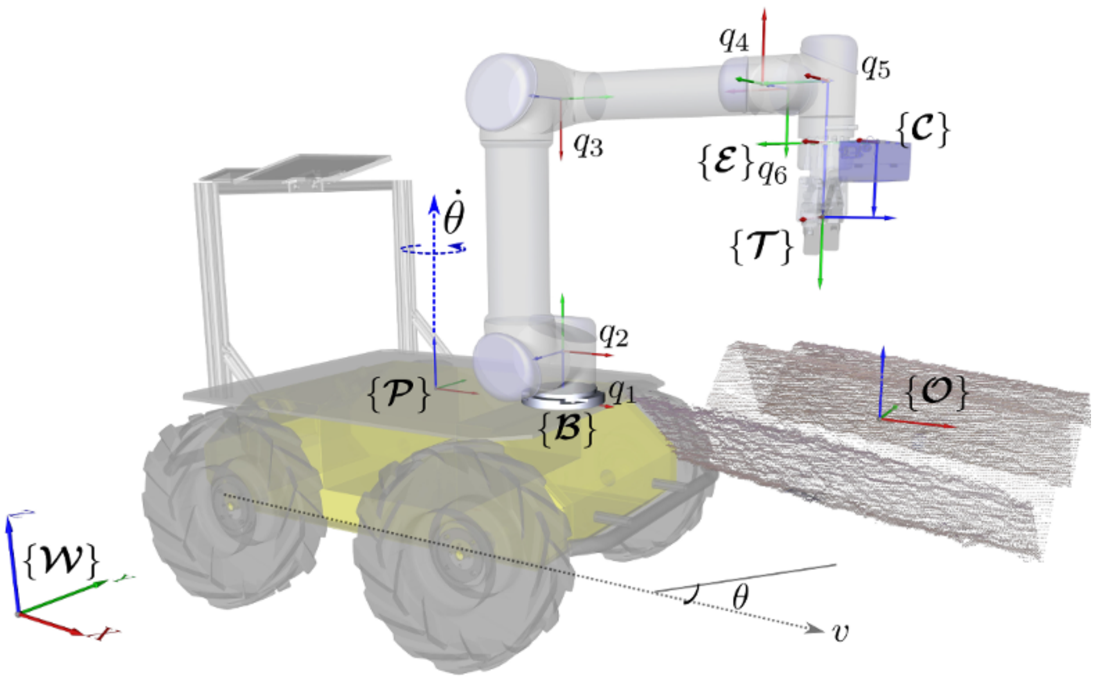


Figure 4.1: Kinematics of the NMM used in this work. Frames in figure are as follows: $\{\mathcal{W}\}$ denotes the global world frame; $\{\mathcal{P}\}$ is the frame associated with mobile platform; $\{\mathcal{B}\}$ is the robot base frame; $\{\mathcal{E}\}$ is the end-effector frame; $\{\mathcal{T}\}$ is the tool frame; $\{\mathcal{C}\}$ is the camera frame; and $\{\mathcal{O}\}$ is the frame associated with the point cloud of the test object.

manipulators (NMMs). They have the shortcoming of adding a coupling at the wheels, which constrains the movements of the platform.

Contribution

The main contributions of this chapter are:

1. We extend our planner [POF$^+$20a] to mobile platforms. We define a cost function which accounts for arm manipulability, path length at the end-effector, and overall motion of the mobile base. Each cost can be accomodate to optimise different performance indices.

2. We incorporate the non-holonomic constraints of the mobile base and the surface contact and alignment constraints at the end-effector into an extended constrained Jacobian. We use the pseudo-inverse of this Jacobian to compute potential robot

configurations.

3. We report experimental results on a simulated Clearpath Husky vehicle equipped with a UR5 manipulator. We compare behaviours obtained with different choices of weights for each cost term. We present experimental results of planning cutting paths on several objects with different surface shapes.

4.2 Mathematical formulation

A cutting task demands a path or a trajectory for the cutting tool. Let's describe the pose of the tool with the function $\mathbf{T_r}(t)$, at each instant $t \in [0, 1]$ (a normalised time variable). $\mathbf{T_r}(0)$ represents the initial tool's pose and $\mathbf{T_r}(1)$ the destination pose. The goal of our algorithm is then to devise a path for $\mathbf{T_r}(t)$, $\forall t \in [0, 1]$. Given a NMM, Fig. 4.1, we consider its configuration vector $\mathbf{q} = [\mathbf{q_p}^T\ \mathbf{q_m}^T]^T$, where $\mathbf{q_p}^T \in \mathbb{R}^{n_p}$ and $\mathbf{q_m}^T \in \mathbb{R}^{n_m}$ are the generalised coordinates of the platform and of the manipulator respectively. The resulting kinematic map to the task variables $\mathbf{r} \in \mathbb{R}^{n_r}$ is

$$\mathbf{r} = \mathbf{f}(\mathbf{q_p},\ \mathbf{q_m}). \tag{4.1}$$

Classically, nonholonomic constraints might be described as

$$\dot{\mathbf{q}}_\mathbf{p} = \mathbf{G}(\mathbf{q_p})\mathbf{u_p}, \tag{4.2}$$

where $\mathbf{u_p} = [\mathbf{v}^T\ \dot{\theta}^T]^T \in \mathbb{R}^{n_v}$ are the inputs of the mobile platform, namely the linear and angular velocity (Fig. 4.1), with $n_v < n_p$, and $\mathbf{G}$ is a $n_p \times n_v$ matrix spanning the admissible velocities at each configuration $\mathbf{q_p}$. On the contrary, the manipulator is an unconstrained system, which poses an equality on its inputs as:

$$\mathbf{u_m} = \dot{\mathbf{q}}_\mathbf{m}. \tag{4.3}$$

Differentiating the kinematic map in Eq. (4.1) over $\mathbf{q}$ and substituting the inputs' vector into the equation, we obtain the extended Jacobian of the robot [DOG07]. In other words, Eq. (4.4) connects the system inputs to the e-e's velocities in the task space.

$$\dot{\mathbf{r}} = \begin{bmatrix} \mathbf{J_p}(\mathbf{q})\mathbf{G}(\mathbf{q_p}) & \mathbf{J_m}(\mathbf{q}) \end{bmatrix} \begin{bmatrix} \mathbf{u_p} \\ \mathbf{u_m} \end{bmatrix} = \hat{\mathbf{J}}(\mathbf{q})\mathbf{u} \tag{4.4}$$

where $\mathbf{u} \in \mathbb{R}^{n_u}$. In this work, we assume the cutting tool is attached to the robot tip with a rigid transformation. Our goal is to find the most appropriate path for the robot end-effector in order to cut the surface of an object perceived as a noisy point cloud.

One of the main contributions of this work is the formulation of the constrained path-planning problem as the minimisation of a multi-objective cost function which accounts for: i) the distance travelled by the robot e-e throughout the path (C_d), ii) the manipulability of the robot along the path (C_w), and iii) the overall movements of the NMM (C_p). We therefore define the cost to be minimised as

$$C(t) = \begin{bmatrix} K_d & K_w & K_p \end{bmatrix} \begin{bmatrix} C_d(t) \\ C_w(\mathbf{q}(t)) \\ C_p(\mathbf{u_p}(t)) \end{bmatrix} = \mathbf{K}\hat{\mathbf{C}}, \; t \in [0,1] \tag{4.5}$$

where $\mathbf{K}$ is a vector of positive coefficients that weighs each cost and $K_d + K_w + K_p = 1$. C_d is the Euclidean distance travelled from $t = 0$ to $t = 1$.

4.2.1 Manipulability cost - C_w

The cost C_w in eq. (4.5) accounts for the manipulability of the robot. As explained in Sec. 3.2.3, the manipulability of a robot assesses its capability to move or apply force in each direction given a configuration. We resort to the already introduced Yoshikawa's index to extract a meaningful scalar value, ω, from the manipulability matrix. We report again that an ω close to zero means a lack of motions (*e.g.* close to a singularity) and high values of ω translate to possibly large movements without endangering the robot.

Similarly to [POF+20a], we define C_w as:

$$C_w(\boldsymbol{q}(\hat{t})) = \frac{1}{\hat{t}} \int_0^{\hat{t}} w\Big(\mathbf{q}(t)\Big)^{-1} \, dt, \tag{4.6}$$

where the reciprocal of the manipulability measure w is integrated over the path, $\hat{t}$ is an instant within the normalised time $[0, 1]$, and t is the integration variable spanning over time.

4.2.2 NMM movement cost - C_p

The object to be cut is visually perceived as a point cloud (PC). A PC is usually affected by noise and can only capture an unaccurate representation of the original object. We assume the surfaces of the objects are infinitely derivable (smoothness), and an inner product exists between two vectors on the surface. Thanks to these assumptions, the surface (or generally the manifold) describes a Riemannian Manifold (RM), Sec. 3.2.4.

Given a point, $\mathbf{p}$, on an RM, the neighbourhood of $\mathbf{p}$ approximates a plane, usually called the tangent plane of the point $\mathbf{p}$. Two functions called *exponential* and *logarithmic* maps connect respectively points on the tangent plane to the manifold and points on the manifold to the tangent plane [LN14]. We further assume that the object captured by the PC retains the properties of the RM. We can then leverage on the *exponential* map and *logarithmic* map to interpolate points on the PC. The plane computed by approximation of a point $\mathbf{p}$ onto the PC represents a bilateral constraint for the robot end-effector, and it may be expressed in the canonical form

$$\mathbf{H}\dot{\mathbf{r}} = \mathbf{0}, \tag{4.7}$$

where $\mathbf{H} \in \mathbb{R}^{n_c \times n_r}$ is a matrix which describes the n_c constraints on the robot e-e pose. Because our goal is to obtain a robot movement that lies on the virtual plane described

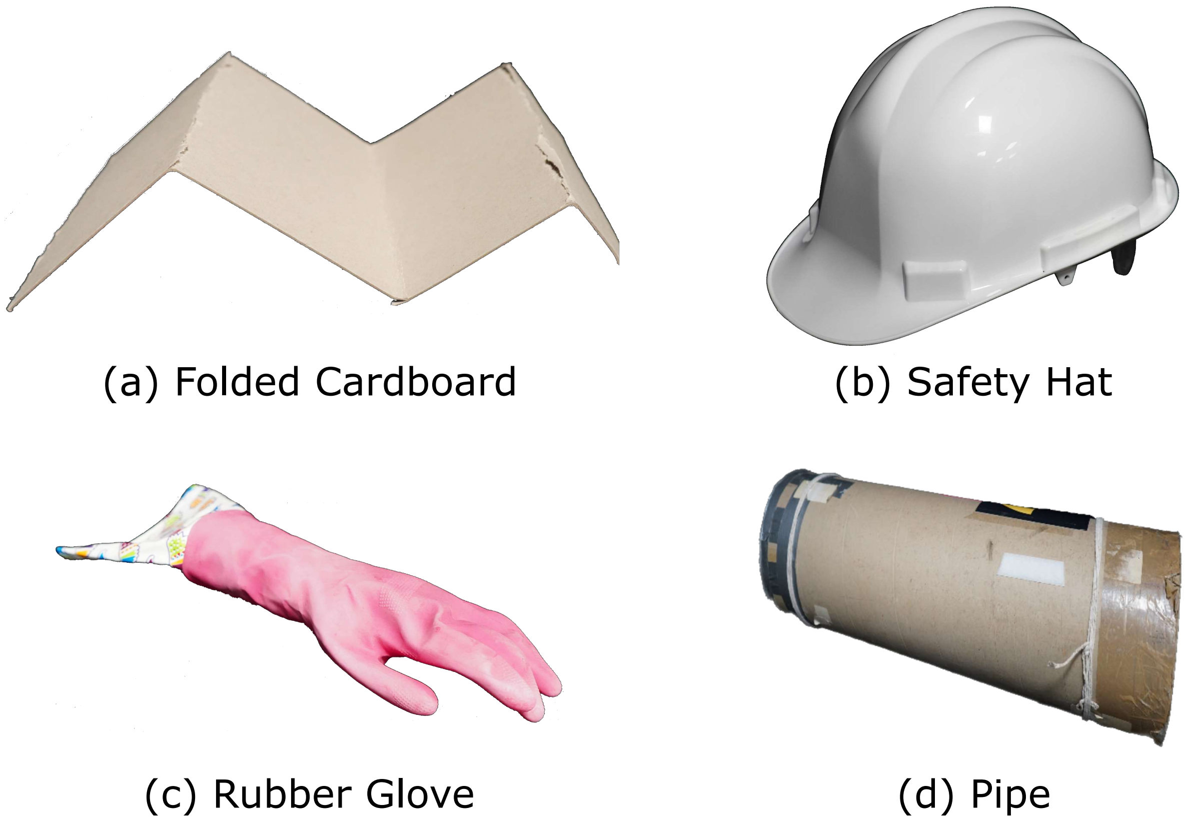


Figure 4.2: Objects used in our experiments. These are good examples of nuclear waste that need cutting.

by the matrix $\mathbf{H}$, we propose to account for the available directions extending Eq. (4.4), and map the motion into the *null-space* of the constraint matrix $\mathbf{H}$.

$$\mathbf{u} = \hat{\mathbf{J}}^{\dagger}(\mathbf{I} - \mathbf{H}^{\dagger}\mathbf{H})\dot{\mathbf{r}} = \mathbf{J}^{\#}\dot{\mathbf{r}} \tag{4.8}$$

The new constrained system has $n_{cs} = n_u - n_c$ degrees of freedom (dofs), where n_c is the number of imposed virtual constraints, to have the robot's e-e stay on the PC surface. An NMM has potentially unlimited workspace due to the mobile base. However, its traversing motions are generally less precise than the arm's and could introduce uncertainties due to improper modelling of the wheels' friction and the potentially uneven terrain. For these reasons, we provide a penalisation of the movements of the platform. Since our robot is redundant w.r.t. the task space ($n_{cs} > n_r$), we use the *null-space* of the constrained Jacobian for achieving a secondary task. Thus, we add a term which pulls the velocity of the platform to zero, as per Eq. (4.9).

$$\mathbf{u} = \mathbf{J}^{\#}\dot{\mathbf{r}} - (\mathbf{I} - \mathbf{J}^{\ddagger}\mathbf{J}^{\#})\mathbf{W}(\hat{\mathbf{u}} - \mathbf{u_0}) \tag{4.9}$$

where, $\mathbf{J}^{\ddagger}$ is the pseudo-inverse of $\mathbf{J}^{\#}$, $\mathbf{W}$ is a positive definite matrix of weights, $\hat{\mathbf{u}} = [\mathbf{u_p}\ \mathbf{0_{n_m}}]$, and $\mathbf{u_0} = \mathbf{0_{n_u \times 1}}$ is the reference input. We propose a cost function that accounts for the platform movements along the path computed as

$$C_p(\mathbf{u_p}(\hat{t})) = \int_0^{\hat{t}} |\mathbf{I_p}\mathbf{u_p}(t)|\ dt, \tag{4.10}$$

where $\mathbf{I_p}$ is the inertia matrix of the platform, and $|.|$ is the L2 norm. This cost accounts for movements along the whole path and increases with bigger motions.

4.3 RRT*-CRMM outline

Our cost $C(t)$ can be used in any sampling-based approach. In this work, we incorporate it within the framework of a very well-known sampling-based algorithm, the Rapidly-exploring Random Tree (RRT), Sec. 3.2.1. Every new node is wired to the tree minimising a given cost function, usually the length between nodes. Its flexibility allows the definition of the node structure based on the application we are targeting. In the most common implementation, it explores the joint-space of the robot and stores a robot's configuration in each node. The optimality of RRT, RRT*, is achieved including a method called *rewiring*. The *rewiring* function checks whether a new point improves the wiring of points already present in the tree and updates those connections accordingly.

Similar to the regular RRT*, we store the robot's configuration in each node. With the knowledge of *forward kinematics*, we can uniquely identify the e-e pose of the arm for each node. Alg. 2 shows the pseudo-code of our RRT*-Constrained Riemannian-Mapping Manipulability (RRT*-CRMM). The tree is initialised with the starting configuration of the robot, and we take two inputs: the maximum number of iterations and the e-

Algorithm 2: RRT*-CRMM

```
input  : The max number of iterations, N, and the destination point, $p_g$
output: The path for the robot in configuration space, $q_{Tr}$, and the e-e path $T_r(t)$
1  initialise the tree, T;
2  while $n \leq N$ or $|p_n - p_g| \geq r_R$ do
3      sample a point into the PC, $p$;
4      search for the nearest point in T, $p_n$;
5      compute the tangent plane to $p$, $T_p$;
6      project $p$ on $T_p$, $p'$;
7      generate the constrained Jacobian, $J^{\#}$;
8      compute displacement, $\delta p$;
9      obtain the new configuration for the robot, $q'$;
10     add movements in the null-space $q_n$;
11     if $q_n$ is collision free then
12         obtain the neighbourhood of $q'$ in T, $N$;
13         compute the cost for the new point, $C$;
14         connect the $q_n$ to the tree;
15         rewiring the nodes $N$ through $q_n$;
16     else
17         discard the point;
18     end
19 end
20 return $q_{Tr}$, $T_r$;
```

e's goal position in Cartesian space. For each iteration, we pick a point, p, within the set of points of the captured PC, *line 3*. Although this procedure seems to reduce the exploration space of the algorithm, the assumption on the object surface of being RM (Sec. 4.2.2) allows a fine interpolation of these limited points. Moreover, this approach has the advantage of exploring only sensible areas and overcoming possible missing spots where the PC failed to represent the surface. Then, we search for the nearest point to p present in the tree, p_n (we use eq. (4.1) to map configurations to poses), *line 4* . We compose the constraints matrix, $\mathbf{H}$, and thus the constrained Jacobian by computing the tangent plane at the point p onto the PC, namely T_p. To use the *exponential map* of the RM, we need to have two points belonging to T_p, we, therefore, geometrically project p_n onto the tangent plane, p'. Now, we compute the displacement between the two points p and p', δp. While the new position displacement is straightforward, the orientation is not. Here, we constrain the z orientation of the e-e to be aligned with the inner normal at the contact point, *lines 5-10.* We compute the new candidate configuration for the robot using the differential kinematics by multiplying the pseudo-inverse of $\mathbf{J}^{\#}$ with the total displacement. The algorithm checks whether the candidate is collision-free or not, *line 11.* Finally, we compute the costs for the new configuration and connect the node with the best parent, *lines 12-14.* To reach optimal performance, we apply the standard rewiring to optimise the nodes already in the tree, *line 15.*

4.4 Simulated experiments and performance analysis

In our experiments, we use a simulated Clearpath Husky, which is a nonholonomic mobile platform, with a UR5 manipulator manufactured by Universal Robots. The 6 dofs manipulator is fixed on the skid-steering mobile platform (*i.e.*, the wheels work in pairs, left-side group and right-side group). A bijective equation binds the linear and angular velocities of the robot ($\mathbf{v}$ and $\dot{\theta}$) to the velocities of the wheel trails. We input to our algorithm: the URDF of the NMM, a point cloud of the object to cut, and the start and goal points of the cutting path, $\mathbf{T_r}(0)$ and $\mathbf{T_r}(1)$ respectively. We use two different 3D sensors to obtain the objects' point clouds, an Astra Orbbec camera and an Ensenso IDS camera. The Astra Orbbec camera is a low-cost camera which produces noisy PCs; while the Ensenso IDS camera provides high-resolution PCs. We calibrate the cameras w.r.t. a global frame, which we use as a reference for both PCs and the robot. We test our approach on a variety of objects (Fig. 4.2) whose PCs are of different sizes, shapes, and noisiness, and we show how our approach overcomes the lack of information and also intentionally placed missing points in the PCs.

4.4.1 Analysis of paths w.r.t. different costs

The cost presented in (4.5) is a linear combination of three functions accounting for travelled distance, manipulability, and platform velocity; each function is weighted using a coefficient spanning between 0 and 1. Generally, we would like to have an optimal

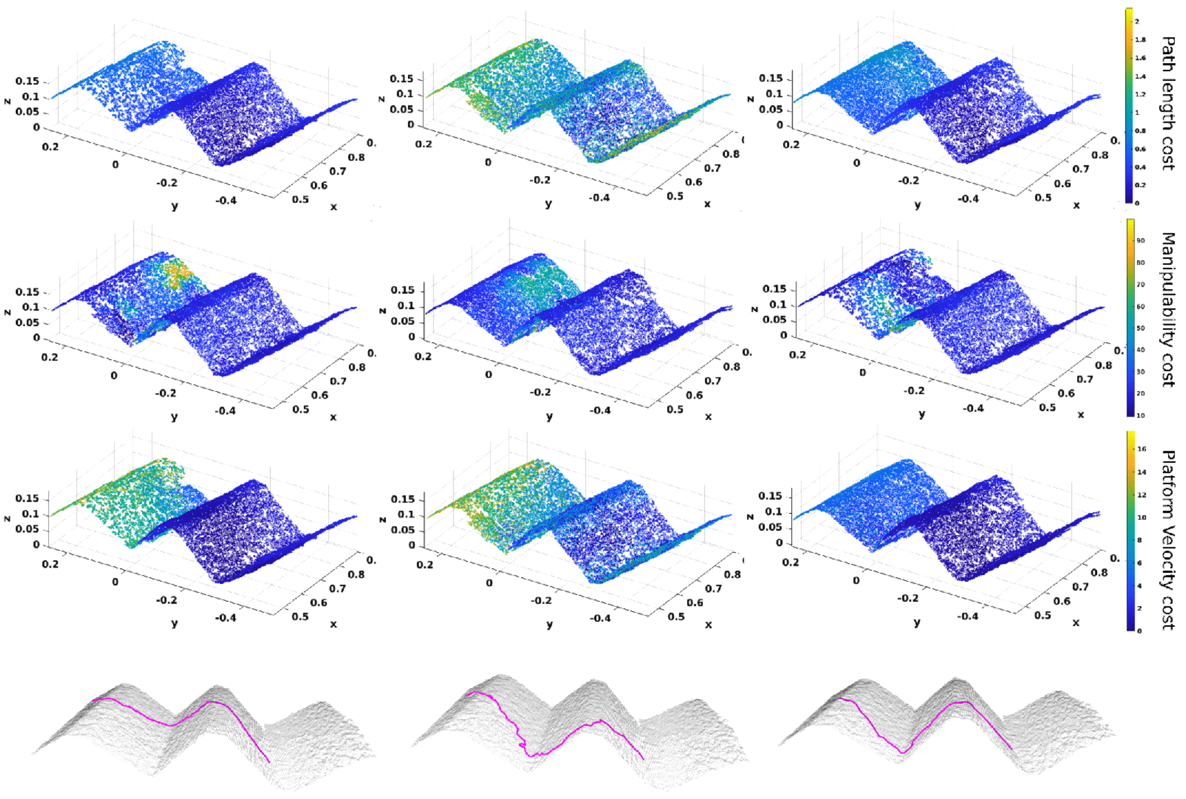


Figure 4.3: This figure reports the effect of the cost weights on the algorithm wiring. Here we show three trials, one for each column, and we set the starting point on the right side of the folded cardboard. Every time, the algorithm runs for 50k iterations, and we set the coefficients to account for only the surface-distance travelled by the e-e (leftmost), only the manipulability of the robot (middle), or only the velocity of the platform (rightmost). For each trial, we report the values obtained by the three objectives. The first line measures the distance cost, the second line the manipulability cost, and the third line the velocity of the base cost. On the right side, a colour map shows the relation between the magnitude of the cost and the colour. We notice that the cost in Eq. (4.6) employees the inverse of the manipulability value, and a lesser cost means a greater manipulability throughout the path from the start. The last line shows the path found in each trial.

triplet of weights applicable to any scenario. However, the intrinsic definition of the problem (*e.g.*, initial pose and configuration of the robot, user or application requirements) demands a case-by-case choice. We analyse the effect of each gain empirically by running our algorithm on the same PC with three sets of gains. Each set singles out the effect of one cost component only by setting to 1 the correlated weight and 0 to the others ($K_D = [1, 0, 0]$, $K_M = [0, 1, 0]$, and $K_P = [0, 0, 1]$, which stand for distance, manipulability, and platform respectively).

In Fig. 4.3, we report the results obtained on the folded cardboard captured with the Astra camera. The first line of the figure shows the surface-distance travelled in the three cases. As we expect, the shortest distances are obtained using K_D (dark blue areas cover all the surface), but a similar result is achieved using K_P as well. K_P retrieves paths with only a small increment in length from K_D, which is visible in the bright blue zone on the left upper ridge. When we single out the manipulability gain, the length consistently increases for reaching every point on the map, and light yellow areas start to appear. The second line of Fig. 4.3 shows the mean manipulability throughout the path in order to reach each point on the surface, and lower values mean higher manipulability as per Eq. (4.6). On the right slope of the cardboard, all three costs obtain good results. However, both K_D and K_P start to decrease the manipulability for reaching points on the left slope of the point cloud. The distance gains particularly struggle in the upper left slope, where it obtains the worse manipulability values (the robot configuration reaches a singularity configuration). Instead, K_P struggles in the lower left part of the cardboard.

The third line shows the platform velocity cost. Small movements of the base over-extend the manipulator with a subsequent reduction of manipulability, and yellow spots appear on the left side of the map. Also, the length of the path for reaching the points on the surface is increased.

The last line shows the optimal paths found for each triplet, using K_D, K_M, and K_P respectively. The leftmost image shows the shortest path between the initial and destination point. The middle one takes a departure from the shortest path for improving

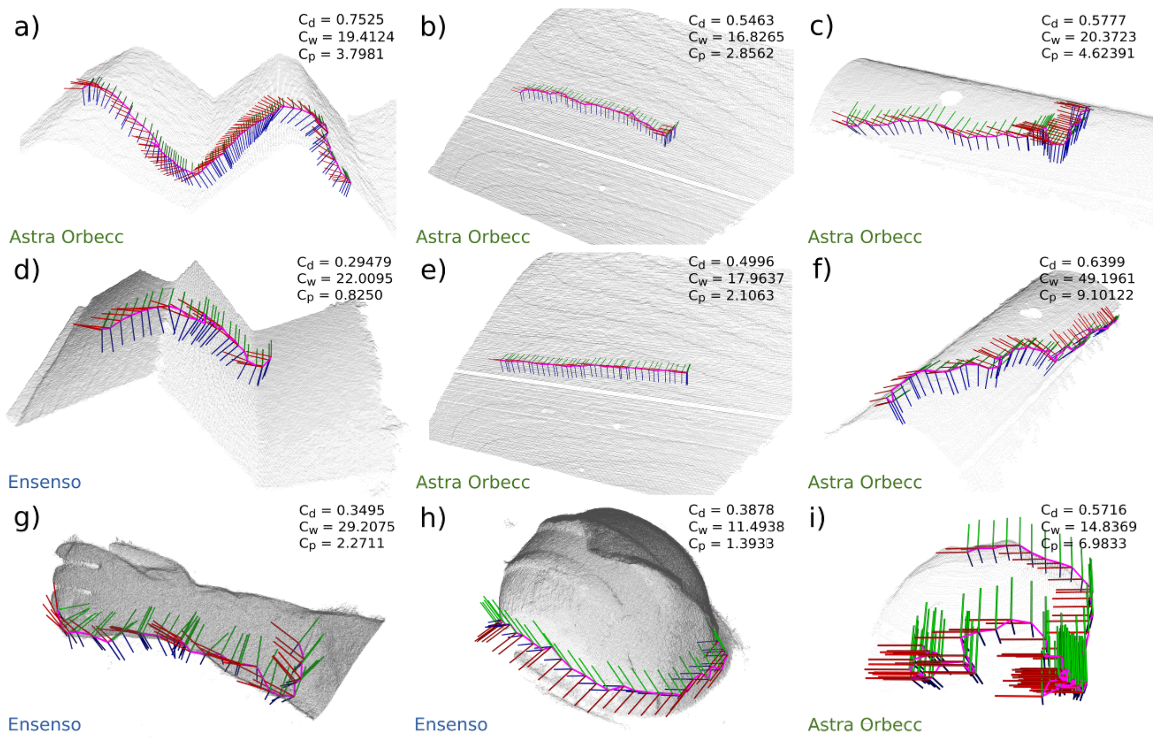


Figure 4.4: This figure shows the results on multiple PCs. In order: a) the folded cardboard captured by the Astra Orbbec; b) flat plane inclined of 30 deg, c) pipe perpendicular to the robot (0 deg rotation), d) folded cardboard captured using the Ensenso; e) flat surface parallel to the ground; f) pipe parallel to the robot; g) glove captured with the Ensenso; h) safety hat captured with Ensenso; i) safety hat captured with Astra Orbbec. On the upper right of each picture, we report the values of each cost component for the optimal path.

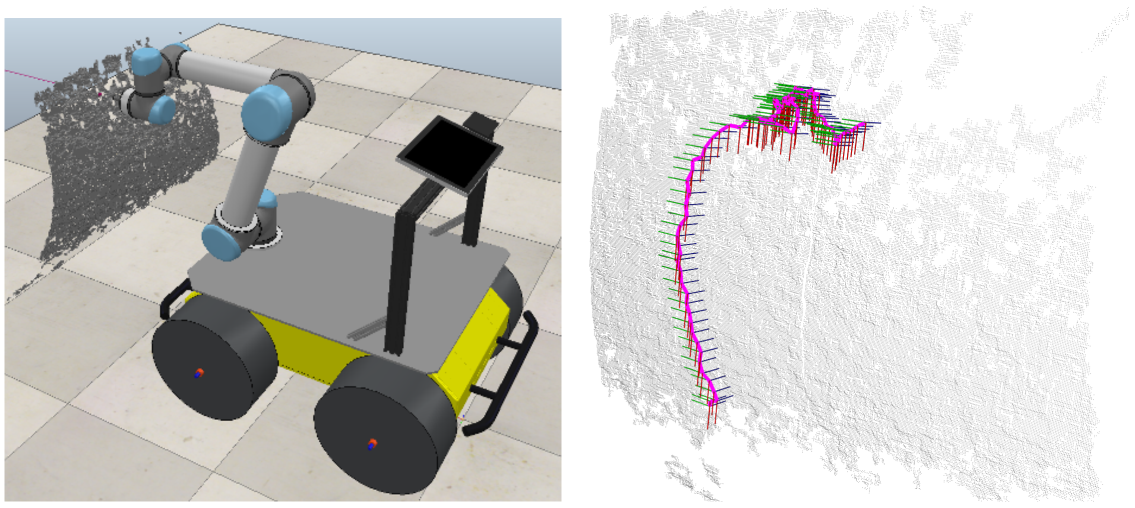

(a) (b)

Figure 4.5: The robot in (a) is tasked with the cutting of the curved object. Because of the relative position between the manipulator base and the object, the robot cannot cut the object in a single operation. On the contrary, our algorithm finds a feasible path as in (b) exploiting the platform dofs.

the manipulability throughout the motion. While a path closer to the lower part of the cardboard is found to reduce the velocity of the platform; this behaviour is consistent with the position of the robot w.r.t. the PC.

4.4.2 Analysis of paths on multiple PCs

We test our algorithm on 5 different objects, Fig. 4.2, which are good examples of typical nuclear wastes with RM surfaces: a plane, a pipe, a safety hat, a plastic glove, and a folded cardboard. We record the PCs for these items using both high- and low-quality cameras, and, before presenting each object to the robot, we shuffle the object's pose.

As we explained in the Sec.4.4.2, the weights impact the algorithm behaviour, and the most appropriate tuning depends on many factors. In Fig. 4.4, we show the best paths found for each PC using the triplet of weights [0.2, 0.5, 0.3], which in our tests provide a good trade-off among the costs over all the scenarios. Every path constrains the robot e-e to lie on the surface of the object and the z-axis of the end-effector to be aligned with the inner normal of the surface at that point. The latter assumption is very common in

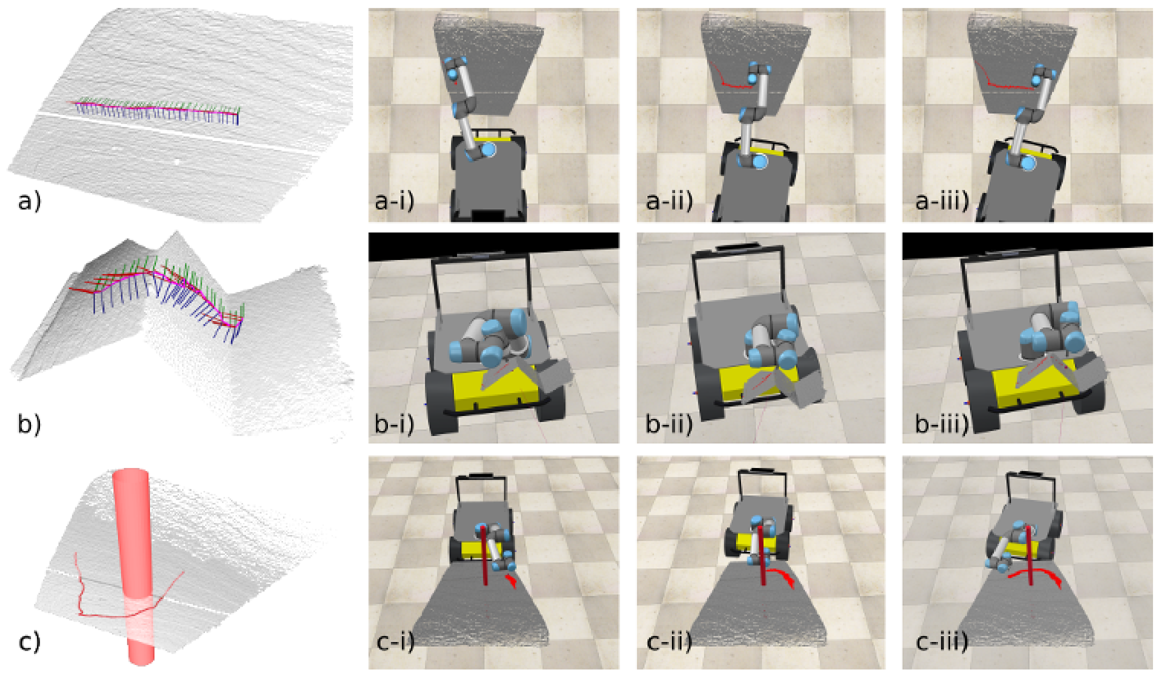


Figure 4.6: Figure of the paths replayed with VREP. For each trial (line) we report: the path generated by our algorithm and 3 screenshots of the robot motions played in VREP.

the cutting operations, where *e.g.* knives or rotatory saws must face towards the object surface. Our algorithm does not need any *a-priori* knowledge of the object and works with arbitrarily shaped PCs. Only for the sake of reporting, the e-e position error forced by the algorithm, we capture the point cloud of a flat plane with a defined height, and we use this number as a ground-truth. Our algorithm successfully finds a path on the reference surface with a mean error of 0.0032 $[m]$ and a standard deviation of 0.0025 $[m]$. All the paths comply with the end-effector pose requirement and follow the surface with the inward normal orientation.

Thanks to the Riemannian assumption on the surfaces, the proposed approach is robust to noise and missing spots in the PCs. Fig. 4.4.c) and 4.4.f) depict a pipe with some missing points in the centre; however, our method finds a solution without avoiding the hole. The implicit smoothness assumption of the real surface allows the algorithm to guess points within the empty area. As we keep the weights constant throughout these tests, a few paths linger on specific areas, *e.g.* Fig. 4.4c) and i). These areas show high manipulability values which are exploited by the algorithm to optimise the overall path. Fine-tuning the gains would solve this issue and provide the robot with a smoother path. We further test the algorithm on surfaces that are not RM, like Fig. 4.4a) and d). Although the object presents discontinuities at the ridges, our approach obtains good results with only a minor approximation of the surface right at the peak of the ridges.

We have already explained how the algorithm leverages the possible departure from the shortest path to improve other indices. Although this increment in length is greatly affected by the weights, we measure the mean ratio between the distance travelled using the weights [0.2, 0.5, 0.3] and the shortest path (obtained with the classical RRT*). In our experiments, we found an average increment of around 30% in path length, a 15% improvement of the manipulability value, and a 17% reduction of the platform base velocity.

In Fig. 4.6, we show a photo-sequence of the trajectories executed by the Husky with UR5 in VREP. The simulation executes the configurations provided by the algorithm in feedforward and draws the path with a red marker. The figure shows how the paths are suitable for driving a complex NMM model.

4.4.3 Comparison between fixed base and mobile base

The cutting of big objects is challenging for fixed-based robots. Usually, the operator must relocate multiple times the object in front of the robot to finish the cutting. NMM systems overcome this problem thanks to platform mobility, and our approach exploits this capability directly in the optimisation phase. In Fig. 4.5(a), we task the robot with the cutting of a curved wall that is very close to the base of the robot. Then, we task the algorithm to find a solution using either standard NMM or a fixed-base arm. In the latter case, we overlooked the C_p cost setting the weight to zero. Our algorithm could not find any solution for the fixed-based robot after 100k iterations but obtains a solution, Fig. 4.5(b), when the motions of the robot are enabled.

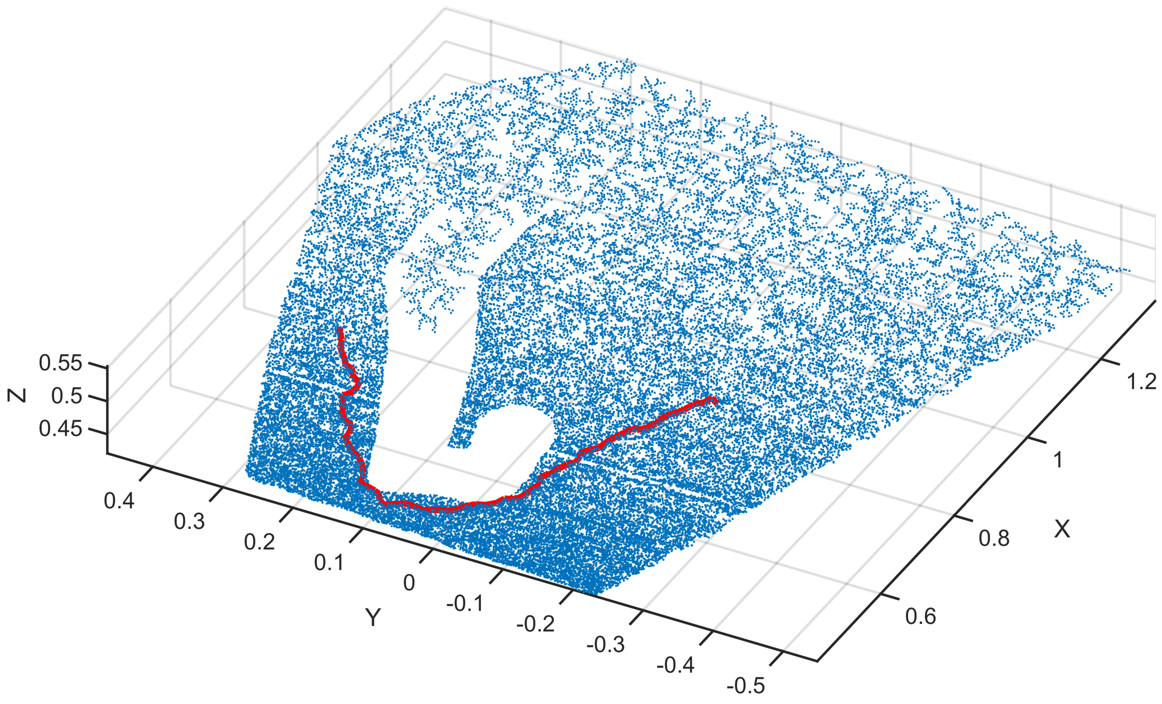


Figure 4.7: The collision-free e-e positions found by RRT*-CRMM are reported using blue lines. Within the white area, the algorithm could not discover any feasible configuration for the NMM, which would not hit the obstacle. In red, we show the proposed path by RRT*-CRMM.

4.4.4 Obstacle avoidance analysis

The previous experiments do not account for obstacles. However, our algorithm may be easily extended to consider obstacle avoidance. The RRT* framework naturally embeds collision avoidance by checking whether a proposed configuration is collision-free or not. In this work, we assume to know beforehand the location of every obstacle in the scene. In Fig. 4.6a the algorithm finds a path connecting the initial and goal points passing through the middle of the PC. However, when we add an obstacle, the robot must find a new route to reach the goal point. In Fig. 4.6c, a new proposed path for the robot e-e follows a path that is clear from the obstacle. Although the discrepancy might seem excessive, the longer path is due to the collision avoidance property of the method, which discards configurations close to the object. In fact, Fig. 4.7 shows how the algorithm rejects points that are not feasible for the manipulator given the task constraints and the arm's configuration. The generated path, in red, is the best path for this scenario.

4.5 Conclusion

This chapter proposes a path planner for an NMM accounting for kinematic constraints on the e-e. Given a point cloud observed by a vision sensor, our algorithm returns a path considering path length, arm manipulability, and mobile base motion. We introduced a novel cost function, which accounts for the surface-distance travelled, the arm manipulability throughout the path, and the velocity of the platform base. Also, we show how to formulate a constrained Jacobian that bounds the robot end-effector to stay in contact with the surface to cut. We incorporated the proposed cost into the well-known RRT* algorithm and analysed the performance. In the experimental section, we show how it is possible to tune the weights in the cost function to elicit different behaviours from the algorithm. Moreover, we quantitatively show that the robot keeps the desired orientation and stays in contact with the object's surface throughout the whole path. Although we assume the objects to be smooth, we perform experiments also on non-Riemannian surfaces, *e.g.* the folded cardboard, to show that our algorithm finds an approximated solution also for non-smooth surfaces.

CHAPTER 5

Motion planning and control of cutting trajectories with a target manipulability ellipsoid

5.1 Introduction

We have shown that manipulability represents a practical method for generating paths robust to uncertainties and singularities. In the previous chapters, we manipulated the original manipulability matrix to obtain a scalar. This approach has the benefit of reducing the conceptual and mathematical effort in assessing the manipulability and allows its straightforward implementation as an objective. However, shrinking information from a matrix to a scalar has the pitfall of discarding valuable information on the considered joint configuration.

If we disjoint the manipulability matrix by singular value decomposition, we obtain three matrices: a diagonal matrix with eigenvalues in descending order and two matrices containing eigenvectors. The combination of eigenvectors and eigenvalues assesses the value of the remaining effort, or velocity, in every direction of the task space. This additional degree of information releases the true value of the manipulability and allows to produce paths with robustness behaviours tailored to the application at hand.

In assembling tasks, the peg-in-hole is fairly common *e.g.* to connect two parts. Generally, one of the two items has a sticking peg, and the other has a bedding hole. The task is to insert the peg into the hole to assemble the final product. The task demands a high precision to centre the peg onto the hole and great flexibility to push the peg in. Many authors have attempted to deploy canonical robots to perform the task but failed to address cases where misalignments between peg and hole were present. Although the manipulator has the power to push the peg, the displacement in orientation creates high frictions, which may reach the breaking point of one of the two objects.

Soft robots and soft actuators compensate for the misalignment, but they are difficult to control and find very little space for deployment in industrial scenarios.

By observation of the task, we may extract some convenient insights. The peg must push on the axis along the rod and adjust its position on the plane perpendicular to the peg's movement. This behaviour increases the likelihood of a successful task against disturbances.

We, therefore, are seeking a robot's trajectory that matches the profile of forces along each axis. The manipulability matrix provides a remarkable tool for addressing this task.

By observation of the task, we may extract some convenient insights. The peg must push on the axis along the rod and adjust its position on the plane perpendicular to the peg's movement. This behaviour increases the likelihood of a successful task against disturbances.

We, therefore, are seeking a robot's trajectory that matches the profile of forces along each axis. The manipulability matrix provides a remarkable tool for addressing this task. It describes the task concisely, and we can use it to generate informed trajectories for the robot.

Here, we also investigate the creation of trajectories for the robot that interact with arbitrary surfaces perceived by noise sensors. Because we consider tasks where the robot must perform full-force operations on the object's surface, we adopt the force manipulability matrix, which hold information about the required task effort.

Contributions

In this chapter, we propose a framework for performing full-force operations that accept a possible departure from the shortest solution to elicit the robustness of the task. Our work aims to plan the best sequence of motions for a manipulator whose manipulability ellipsoid throughout the trajectory matches a *task-informed* ellipsoid, which is designed by the model interaction of the task. We then constrain the robot's end-effector to stay on a surface, which is sensed by a noisy sensor. We call our approach TEMPO. As far as we know, no other work yields a trajectory which jointly lies on a noisy Point Cloud and minimises the distance between the current manipulability and the *task-informed* ellipsoid along the entire path. Contrary to Chapters 3 and 4, where we have implemented gradient-free path planners, here we can compute the gradient of the objective functional to drive the optimisation procedure. Therefore, we frame the problem as an optimisation-based path planner. We also leverage the Hybrid Force/Motion controller and the impedance controller to develop a control strategy that embodies the feature of the generated trajectory and can react to unforeseen uncertainties.

The contributions of this paper are:

1. We propose a novel objective function that accounts for the shape of the manipulability ellipsoid over the trajectory;

2. We design an optimisation problem using the concept of functionals, which minimises the entire trajectory;

3. We introduce a methodology to define and incorporate the constraints at the end-effector to lie on a surface, which is provided by a noisy sensor;

4. We incorporate a compliant behaviour within the motion space of the Hybrid

Force/Motion Controller;

5. We tested our algorithm, together with the proposed controller, on a real scenario with various objects.

5.2 Trajectory optimisation formulation

We propose a constrained multi-objective optimisation of the entire trajectory, which may be formulated as per eq. (5.1).

$$\begin{aligned} \min_{\boldsymbol{\xi}} \quad & \mathcal{U}[\boldsymbol{\xi}] \\ \text{s.t.} \quad & \mathcal{H}[\boldsymbol{\xi}] = 0 \\ & \boldsymbol{\xi}[t_0] = \boldsymbol{\xi}_0 \\ & \boldsymbol{\xi}[t_F] = \boldsymbol{\xi}_{K+1} \end{aligned} \tag{5.1}$$

where $\boldsymbol{\xi} \in \mathbb{R}^{Kn}$ is a vector representing K way-points of the trajectory, with each element a joint configuration of the robot (with size n), thus $\boldsymbol{\xi} = [\boldsymbol{\xi}_1^T \dots \boldsymbol{\xi}_K^T]^T$. Our objective function, $\mathcal{U}[\boldsymbol{\xi}]$, must account for the smoothness of the robotic movements and minimise the similiraties between the e-e's manipulability ellipsoid and the task-informed ellipsoid:

$$\mathcal{U}[\boldsymbol{\xi}] = \lambda \mathcal{F}_{smooth}[\boldsymbol{\xi}] + \mathcal{F}_{man}[\boldsymbol{\xi}] \tag{5.2}$$

where $\mathcal{F}_{smooth}[\boldsymbol{\xi}]$ is the objective accounting for the smoothness of the trajectory, $\mathcal{F}_{man}[\boldsymbol{\xi}]$ accounts for the distance between manipulability ellipsoids, and $\lambda \in \mathbb{R}$ is a coefficient to weigh the two objectives. Moreover, the optimisation is subject to the constraint on the e-e' pose, which must lie on the object's surface throughout all the trajectory. The equation $\mathcal{H}[\boldsymbol{\xi}]$ is a canonical representation of the constraint, which requires the displacement between the robot's e-e and the surface to be zero.

Because we have no prior information about the object's shape, a mathematical rep-

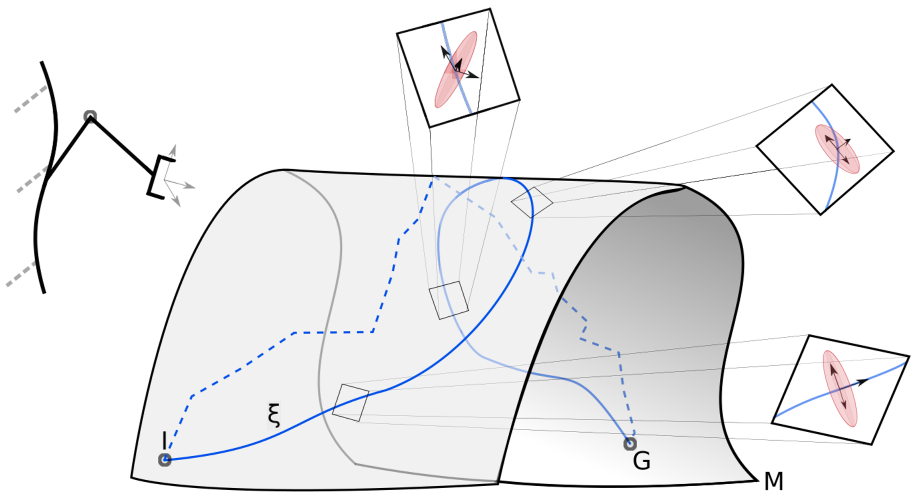


Figure 5.1: A trajectory $\boldsymbol{\xi}$ connects point I and G onto the manifold M. The underlying surface constraints the robot's motions. Along the path, we require the robot to have a task-dependant manipulability ellipsoid.

resentation of the surface is not available, and we perceive the object through a noisy PC. Therefore, the equation $\mathcal{H}[\boldsymbol{\xi}]$ needs to be locally estimated using one of the available methods in the literature for each way-point; the Principal Component Analysis (PCA) is a technique for this purpose when the implicit surface is smooth such as our assumptions.

5.2.1 Manipulability

In literature, there are multiple definitions of manipulability based on the applications to target *i.e.* the velocity ellipsoid, the force/torque ellipsoid, and the dynamic manipulability, [Yos85b, DSMB95, Yos85a]. Hereafter, we describe an alternative formulation for the manipulability from the one in Sec. 3.2.3 that addresses torques rather than velocities.

We constrain the vector of torques for a given configuration to have a unitary norm. The obtained matrix, $\boldsymbol{M}$, maps the torque $\boldsymbol{\tau}$ to the forces $\boldsymbol{F}$, as per eq. (5.3).

$$|\boldsymbol{\tau}| = \boldsymbol{\tau}^T\boldsymbol{\tau} = \boldsymbol{F}^T\boldsymbol{J}(\boldsymbol{q})\boldsymbol{J}(\boldsymbol{q})^T\boldsymbol{F} = \boldsymbol{F}^T\boldsymbol{M}(\boldsymbol{q})\boldsymbol{F} = 1 \tag{5.3}$$

This matrix is called *Manipulability matrix*, and several work have exploited this concept to *e.g.* avoid kinematic singularities [KMCY04, STMPG17], improve the control strategy [SLM+19], and for planning [US03, NHGT02].

The singular value decomposition (SVD) of $\boldsymbol{M}$ generates three matrices, $\boldsymbol{U}$, $\boldsymbol{\Phi}$, and $\boldsymbol{V}$:

$$\boldsymbol{M} = \boldsymbol{U}\boldsymbol{\Phi}\boldsymbol{V}^T \tag{5.4}$$

The matrix $\boldsymbol{\Phi}$ is a diagonal matrix whose elements are the eigenvalues of $\boldsymbol{M}$, and the matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ are unitary matrices that map torques to forces and *viceversa*. Because of the definition, $\boldsymbol{M}$ shapes an ellipsoid in the operational space whose axis represents the residual robot's efforts in each direction, and the eigenvalues scale the axis.

We are interested in measuring the distance between two manipulability ellipsoids described by their matrices. In general, the operations are not straightforward between matrices.

A few well-known methods propose to reduce the problem and consider either the square root of the eigenvalues [Yos85b] or the inverse of the conditioning number [WW99]. These simplifications yield a convenient measure for manipulability, but they fail to consider the directions of the ellipsoid.

Conversely, other approaches assess the similarities between two matrices as a whole (*e.g.* the Frobenius norm or L_p norm), but they are usually very computationally expensive and are not suitable for iterative optimisation algorithms. We, therefore, need to exploit the inherent properties of the problem to select a better metric that is more computationally efficient.

In [RJCC17] and [JRCC0], the authors notice that the manipulability matrices are Symmetric Positive Definite (SPD) by definition. Hence, they belong to the Riemannian manifold of SPD matrices, and we can exploit a tailored metric for this manifold.

5.2.2 Riemannian Manifold of SPD matrices

In Sec. 3.2.4, we introduced the Riemannian Manifold (RM) as a smooth l-topological space that approximates the Euclidean space in the neighbourhood of its points.

The Riemannian Geometry allows to define two maps *exponential* map and *logarithmic* map. These mappings moves points onto the manifold $\mathcal{M}$ to the tangent plane $T_p\mathcal{M}$ and *vice-versa*, Fig. 3.3. Besides our formulation to map points from the PC to the object surface, the RM extends to a broad range of manifolds spanning from spheres to matrix sets.

Let $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{H}_{\mathbb{D}}$ be two Symmetric and Positive Definite matrices, where $\mathbb{H}_{\mathbb{D}}$ is the set of $\mathbb{D} \times \mathbb{D}$ Hermitian matrices. We denote the set of SPD matrices as $\mathbb{P}^+$. $\mathbb{P}^+$ is a well-studied differentiable RM with the Riemannian metric given by the differential form $ds = ||\boldsymbol{A}^{-1/2} d\boldsymbol{A}\boldsymbol{A}^{-1/2}||_F$, with the F standing for the Frobenius norm. It yields the Riemannian distance in eq. (5.5) which uses the matrix logarithm [Bha07].

$$\delta_R(\boldsymbol{A}, \boldsymbol{B}) = || \log(\boldsymbol{A}^{-1/2}\boldsymbol{B}\boldsymbol{A}^{-1/2})||_F \quad for\, \boldsymbol{A}, \boldsymbol{B} > 0 \tag{5.5}$$

However, the computation of δ_R is time-consuming, and [Sra15] introduces a computationally cheaper approach for obtaining the distance between $\boldsymbol{A}$ and $\boldsymbol{B}$, called Stein Divergence (S-Divergence), eq. (5.6).

$$\delta_S^2(\boldsymbol{A}, \boldsymbol{B}) = \log\left(\det\left(\frac{\boldsymbol{A}+\boldsymbol{B}}{2}\right)\right) - \frac{1}{2}\log(\det(\boldsymbol{A}\boldsymbol{B})) \tag{5.6}$$

Such distance is not a metric; however, it is non-negative, definite and symmetric. Nonetheless, Sra demonstrates in [Sra15] how the square of the S-Divergence is a metric for the set $\mathbb{P}^+$. Thanks to its fast computation, as is its derivative, it is more suitable for iterative optimisation algorithms (more details on this metric in [Sra15]).

5.2.3 Objective functional

At every iteration, we change the entire trajectory to minimise a multi-objective function. Because we are accounting for the complete path at every iteration, we talk of an objective functional.

Every cost is formulated using an integral over the entire trajectory, such as:

$$\int_0^1 v(\boldsymbol{\xi}(t), \boldsymbol{\xi}'(t))\ dt \tag{5.7}$$

where $\boldsymbol{\xi}'$ is the time derivative of the $\boldsymbol{\xi}$, v is the function to optimise, and the trajectory is normalised between 0 and 1.

We introduce a novel objective function, $\mathcal{F}_{man}$, which measures the distance between all manipulability matrices throughout the trajectory from the task-informed matrix, $\hat{\boldsymbol{M}}$, and sums up all these contributions.

$$\mathcal{F}_{man}[\boldsymbol{\xi}] = \int_0^1 g(\boldsymbol{\xi}(t), \hat{\boldsymbol{M}})dt \tag{5.8}$$

Eq. (5.8) leverages on the square of the S-Divergence metric to compute the distance between two SPD matrices, such as:

$$g(\boldsymbol{\xi}(t), \hat{\boldsymbol{M}}) = \sum_{k=1}^{K} \delta_s(\boldsymbol{M}(\boldsymbol{\xi}_k(t)), \hat{\boldsymbol{M}}) \tag{5.9}$$

We might define the target manipulability matrix by leveraging the physical interaction models, which are task-dependent and increase the capacity of the robot to promptly react to uncertainties. For instance, [MXJ19] studies the cutting of vegetables with a knife and proposes an interaction model object/tool. Although the task-informed ellipsoid must belong to $\mathbb{P}^+$, it describes properties in Cartesian space and does not require any information about the kinematic of the robot. Hence, it is straightforward to generate the proper ellipsoid depending on the time.

As for the smoothing objective, we adopt the equation proposed in [RZBS09] and

[ZRD+13] which minimises the movements of the trajectory's way-points:

$$\mathcal{F}_{smooth}[\boldsymbol{\xi}] = \frac{1}{2}\int_0^1 \left\| \frac{d}{dt}\boldsymbol{\xi}(t) \right\|^2 dt \tag{5.10}$$

Zucker et al. [ZRD+13] also introduce a third objective, which takes care of generating a collision-free trajectory for the arm. In this work, we overlook the obstacle avoidance problem because the intrinsic structure of the application that requires an interaction with the environment and obstacles may be usually neglected. However, our discussion does not pose any limitations on using more costs, and an objective accounting for the collisions might be introduced back without further analysis.

5.2.4 Optimisation method

A big body of the literature study how to solve the minimisation problem in eq. (5.1). In this chapter, we frame the optimisation similarly to [RZBS09], and we set up the objective functionals to be suitable for the functional minimisation. We will solve the free optimisation first and show how to constrain the problem afterwards.

Because we parametrised the trajectory, $\boldsymbol{\xi}$, by a series of K way-points, we write the smoothness objective, eq. (5.10), as the sum of a finite differentiating matrix $\boldsymbol{Z}$ and a vector $\boldsymbol{e}$, and consequently, we can shape the equation using a quadratic form:

$$\mathcal{F}_{smooth}[\boldsymbol{\xi}] = \frac{1}{2}||\boldsymbol{Z}\boldsymbol{\xi} + \boldsymbol{e}||^2 = \frac{1}{2}\boldsymbol{\xi}^T\boldsymbol{\Delta}\boldsymbol{\xi} + \boldsymbol{\xi}^T\boldsymbol{\beta} + \gamma \tag{5.11}$$

with $\boldsymbol{\Delta} = \boldsymbol{Z}^T\boldsymbol{Z}$, $\boldsymbol{\beta} = \boldsymbol{Z}^T\boldsymbol{e}$, $\gamma = \boldsymbol{e}^T\boldsymbol{e}/2$, and $\boldsymbol{Z}$ an $\boldsymbol{e}$ are:

$$\boldsymbol{Z} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 & 0 \\ -1 & 1 & 0 & \dots & 0 & 0 & 0 \\ 0 & -1 & 1 & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & -1 & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & -1 & 1 \\ 0 & 0 & 0 & \dots & 0 & 0 & -1 \end{bmatrix} \otimes \boldsymbol{I}_{m\times m}$$

$$\boldsymbol{e} = [-\boldsymbol{\xi}_0, 0, \dots, 0, -\boldsymbol{\xi}_{K+1}]^T$$

where $\boldsymbol{\xi}_0$ and $\boldsymbol{\xi}_{K+1}$ are the starting and the goal points of the trajectory respectively, and $\otimes$ denotes the Kronecker product between a matrix of size $K+1 \times K$ and the identity matrix belonging to $\mathbb{R}^{m\times m}$.

We want to use the steepest descent algorithm to search for the optimal trajectory according to the cost functional. We, therefore, are asking for the optimisation method to obtain the trajectory that minimises the quantity:

$$\boldsymbol{\xi}_{i+1} = arg\ \underset{\boldsymbol{\xi}}{min} \left\{ \mathcal{U}[\boldsymbol{\xi}] + \delta\boldsymbol{\xi}^T \nabla\mathcal{U}[\boldsymbol{\xi}] + \frac{\eta}{2}||\delta\boldsymbol{\xi}||^2_{\boldsymbol{\Delta}} \right\} \tag{5.12}$$

where η is the step size, $\delta\boldsymbol{\xi} = \boldsymbol{\xi} - \boldsymbol{\xi}_i$ is the displacement between trajectories, and the notation $||\delta\boldsymbol{\xi}||^2_{\Delta}$ represents the norm of the displacement w.r.t. the Riemannian Manifold $\boldsymbol{\Delta}$, namely $\delta\boldsymbol{\xi}^T\boldsymbol{\Delta}\delta\boldsymbol{\xi}$, [ZRD+13].

Deriving eq. (5.12) and setting the result to zero, we finally obtain the updating rule:

$$\boldsymbol{\xi}_{i+1} = \boldsymbol{\xi}_i - \frac{1}{\eta}\boldsymbol{\Delta}^{-1}\nabla\mathcal{U}[\boldsymbol{\xi}] \tag{5.13}$$

At each iteration, the updating rule needs to compute the gradient of the cost functional $\mathcal{U}[\boldsymbol{\xi}]$. Stemming from the functional mathematics, this gradient, $\nabla\mathcal{U}[\boldsymbol{\xi}]$, can be computed as per eq. (5.14).

$$\nabla\mathcal{U}[\boldsymbol{\xi}] = \frac{\partial v}{\partial \boldsymbol{\xi}} - \frac{d}{dt}\frac{\partial v}{\partial \boldsymbol{\xi}'} \tag{5.14}$$

5.2.5 Gradient of the objective functional

To compute the gradient of eq. (5.2), we can exploit the sum rule of derivative, computing the $\nabla\mathcal{F}_{man}[\boldsymbol{\xi}]$ and $\nabla\mathcal{F}_{smooth}[\boldsymbol{\xi}]$ separately.

The $\nabla\mathcal{F}_{smooth}[\boldsymbol{\xi}]$ gradient is straightforward, and results in taking the opposite direction of the trajectory acceleration, $-\frac{d^2}{dt^2}\boldsymbol{\xi}(t)$.

On the contrary, the gradient of the manipulability term needs a more careful discussion. Since the manipulability objective functional is already in the canonical form presented in (5.7), we can directly use eq. (5.14), having g in place of v. We neglect the second term since the g does not depend on $\boldsymbol{\xi}'$, and use the derivative sum property again to write eq. (5.15).

$$\nabla\mathcal{F}_{man}[\boldsymbol{\xi}] = \frac{\partial\, g(\boldsymbol{\xi}(t))}{\partial\, \boldsymbol{\xi}} = \sum_{k=1}^{K} \frac{\partial\, \delta_s(\boldsymbol{M}(\boldsymbol{\xi}_k))}{\partial\, \boldsymbol{\xi}} \tag{5.15}$$

Each element of the resulting sum is the derivation of a scalar by a vector, and considering the k-th element, we obtain:

$$\frac{\partial\, \delta_s(\boldsymbol{M}(\boldsymbol{\xi}_k))}{\partial\, \boldsymbol{\xi}} = \left[\frac{\partial\, \delta_s(\boldsymbol{M}(\boldsymbol{\xi}_k))}{\partial\, \boldsymbol{\xi}_1} \cdots \frac{\partial\, \delta_s(\boldsymbol{M}(\boldsymbol{\xi}_k))}{\partial\, \boldsymbol{\xi}_K}\right]^T \tag{5.16}$$

The vector in eq. (5.16) has only the k-th entry non-zero. Therefore, we need to compute only this derivative to obtain the entire solution:

$$\frac{\partial\, \delta_s(\boldsymbol{M}(\boldsymbol{\xi}_k))}{\partial\, \boldsymbol{\xi}_k} = \frac{\delta_s^{-1}(\boldsymbol{M}(\boldsymbol{\xi}_k))}{2} \frac{\partial\, \delta_s^2(\boldsymbol{M}(\boldsymbol{\xi}_k))}{\partial\, \boldsymbol{\xi}_k} \tag{5.17}$$

Let us define the matrices $\boldsymbol{C}' = \frac{\hat{\boldsymbol{M}}+\boldsymbol{M}(\boldsymbol{\xi}_k)}{2}$ and $\boldsymbol{C}'' = \hat{\boldsymbol{M}}\boldsymbol{M}(\boldsymbol{\xi}_k)$ from eq. (5.6) and derive the second member of eq. (5.17).

$$
\begin{aligned}
\frac{\partial\, \delta_s^2(\boldsymbol{M}(\boldsymbol{\xi}_k))}{\partial\, \boldsymbol{\xi}_k} =& \frac{1}{det(\boldsymbol{C}')} tr\Big(adj(\boldsymbol{C}')\frac{\partial\, \boldsymbol{C}'}{\partial\, \boldsymbol{\xi}_k}\Big) \\
&- \frac{1}{2}\frac{1}{det(\boldsymbol{C}'')} tr\Big(adj(\boldsymbol{C}'')\frac{\partial\, \boldsymbol{C}''}{\partial\, \boldsymbol{\xi}_k}\Big)
\end{aligned}
\tag{5.18}
$$

where we apply the Jacobi's formula to derive the determinants of the matrices, $tr(.)$ is the trace of the matrix in the parenthesis, and $adj(.)$ stands for the adjoint function of a matrix. Finally, we obtain the derivation of the two matrices, $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$, as:

$$
\frac{\partial\, \boldsymbol{C}_1}{\partial\, \boldsymbol{\xi}_k} = \frac{1}{2}\frac{\partial\, \boldsymbol{M}(\boldsymbol{\xi}_k)}{\partial\, \boldsymbol{\xi}_i} \tag{5.19}
$$

$$
\frac{\partial\, \boldsymbol{C}_2}{\partial\, \boldsymbol{\xi}_k} = \hat{\boldsymbol{M}}\frac{\partial\, \boldsymbol{M}(\boldsymbol{\xi}_k)}{\partial\, \boldsymbol{\xi}_k} \tag{5.20}
$$

Eq. (5.19)-(5.20) need to further derive the manipulability matrix $\boldsymbol{M}$ by the vector $\boldsymbol{\xi}_k$. However, this matrix is not uniquely defined and depends on which manipulability ellipsoid to use, *e.g.*, the force manipulability ellipsoid would be defined using $\boldsymbol{M} = \boldsymbol{J}\boldsymbol{J}^T$ as explained in Sec. 5.2.1.

5.2.6 Local Optimisation Constraints

Thus far, we presented a trajectory optimisation that works for unconstrained scenarios. However, in many cases, we need to pose constraints either on admissible values, *e.g.* the joints' values, or on the properties of the trajectory, *e.g.* a fixed orientation of the e-e.

We may overcome this issue by describing any constraints in the form of a nonlinear differentiable vector function in the Hilbert space of the trajectories, for which $\mathcal{H}[\boldsymbol{\xi}] = 0$ whether $\boldsymbol{\xi}$ satisfies the condition or non-zero otherwise.

The introduction of these constraints in the optimisation algorithm modifies the up-

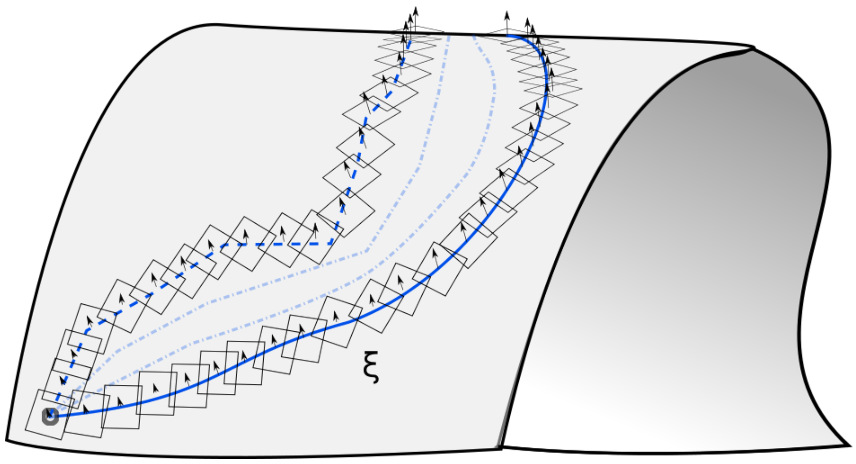


Figure 5.2: The figure shows two examples local optimisation constraints as approximating planes onto a curved surface (grey object). The vectors' origin is onto the object, and they point outward the tangent plane computed at the origin point. The dashed lines indicate two more possible middle states of the optimisation process.

dating rule, which becomes:

$$\begin{aligned}\boldsymbol{\xi}_{i+1} &= \boldsymbol{\xi}_i - \frac{1}{\eta}\boldsymbol{\Delta}^{-1}\nabla\mathcal{U}[\boldsymbol{\xi}] \\ &\quad + \frac{1}{\eta}\boldsymbol{\Delta}^{-1}\boldsymbol{C}^T(\boldsymbol{C}\boldsymbol{\Delta}^{-1}\boldsymbol{C}^T)^{-1}\nabla\mathcal{U}[\boldsymbol{\xi}] \\ &\quad - \boldsymbol{\Delta}^{-1}\boldsymbol{C}^T(\boldsymbol{C}\boldsymbol{\Delta}^{-1}\boldsymbol{C}^T)^{-1}\boldsymbol{b}\end{aligned}$$

where the first line is the same as the unconstrained optimisation in eq. (5.13), and the second and third lines project the unconstrained step the on the Taylor's first order expansion of the constrain $\mathcal{H}[\boldsymbol{\xi}] = 0$ nearby the i-th trajectory $\boldsymbol{\xi}_i$, namely $\mathcal{H}[\boldsymbol{\xi}]|_{\boldsymbol{\xi}_i} \approx \mathcal{H}[\boldsymbol{\xi}_i] + \frac{\partial}{\partial \boldsymbol{\xi}_i}\mathcal{H}[\boldsymbol{\xi}_i](\boldsymbol{\xi}-\boldsymbol{\xi}_i) = \boldsymbol{C}(\boldsymbol{\xi}-\boldsymbol{\xi}_i)+\boldsymbol{b}$, correcting also the offset between the approximated hyperplane and the real constraint (more information in [ZRD+13]).

We need to constrain the tool to touch the object's surface throughout the execution of the task, but we only have a representation of the shape given by a noisy PC. We, therefore, assume the surface to be smooth and locally approximating a plane. Thanks to this assumption, we can think the constraint as a sequence of tangential planes, one for each way-point, and the PCA method provides the 4-tuple $[n_x, n_y, n_z, o] = [\boldsymbol{n}, o]$ describing each plane., Fig. 5.2.

Thanks to this assumption, at each optimisation iteration, the algorithm must compose the matrix $(K \times m)$-size matrix $\boldsymbol{C}$ and K-size vector $\boldsymbol{b}$ by the juxtaposition of the first order approximation of these planes. Therefore, given the k-th way-point, $\bar{\boldsymbol{w}}$, of a trajectory $\boldsymbol{\xi}$, the k-th rows of $\boldsymbol{C}$ and $\boldsymbol{b}$ are:

$$\begin{aligned} \boldsymbol{C}_k &= \boldsymbol{n} \\ \boldsymbol{b}_k &= \boldsymbol{n}^T \bar{\boldsymbol{w}} + o \end{aligned} \tag{5.21}$$

This approach can be successfully used if the object's surface is smooth, which is for many industrial and domestic cases (barrels, battery cells, etc.). Moreover, an extension to non-smooth surfaces might be achieved by running the optimisation multiple times over portions of the surface where it may be considered smooth locally.

5.3 A compliant hybrid force/motion controller

The path provided by TEMPO, $\boldsymbol{\xi}^*$, engenders the optimal robot motion to perform the task. It attains the requirements embedded using the *task-informed* ellipsoid, which uses prime knowledge about the physics involved in the task.

In this work, we address the problem of resizing an object for dismantling purposes, and we, therefore, ask the tool to continuously apply a force normal to the object's surface while the motion is orthogonal.

5.3.1 Hybrid Force/Motion Control

By definition of the problem, the two commands are then orthogonal to one another, and Khatib [Kha88] formulates the so-called *Operational Space Control* theory, which exploits this structure to generate decoupled actions in the task-space.

Let us introduce the dynamics equation of the manipulator in the operational space:

$$\boldsymbol{\Lambda}(\boldsymbol{q})\ddot{\boldsymbol{r}} + \boldsymbol{\mu}_r(\dot{\boldsymbol{q}}, \boldsymbol{q})\dot{\boldsymbol{r}} + \boldsymbol{p}_r(\boldsymbol{q}) = \boldsymbol{F} \tag{5.22}$$

where $\dot{\boldsymbol{r}}$ and $\ddot{\boldsymbol{r}}$ are the first and second order derivative of the end-effector pose, $\boldsymbol{r}$, $\boldsymbol{\Lambda} \in \mathbb{R}^{m \times m}$ is the inertia matrix of the robot, $\boldsymbol{\mu}_r \in \mathbb{R}^m$ accounts for centrifugal and Coriolis terms, $\boldsymbol{p}_r \in \mathbb{R}^m$ is the gravity term, and $\boldsymbol{F}$ represents the generalised forces at the end-effector.

We want to split the vector $\boldsymbol{F}$ into two independent forces, $\boldsymbol{F}_m$ and $\boldsymbol{F}_a$; the former accounts for the motion parallel to the surface, and the latter take care for the action forces pointing inward to the surface.

A convenient approach to define these forces is by the definition of a diagonal matrix, whose main diagonal is composed of 0s and 1s whether a movement in that direction is permitted or not. Thus, we define a matrix $\boldsymbol{\Sigma}_f$ that defines the available movements in the operational space for the robot's e-e:

$$\boldsymbol{\Sigma}_f = \begin{bmatrix} \sigma_x & 0 & 0 \\ 0 & \sigma_y & 0 \\ 0 & 0 & \sigma_z \end{bmatrix} \tag{5.23}$$

where $\sigma_x, \sigma_y, \sigma_z \in \{0, 1\}$. Moreover, the dual matrix of available force directions is obtained using $\tilde{\boldsymbol{\Sigma}}_f = \boldsymbol{I}_{3\times3} - \boldsymbol{\Sigma}_f$.

Since we want the e-e's frame and the described matrix match, we need to compute the rotational matrix $\boldsymbol{S}_f(t)$, which aligns the two frames at each instant. Applying this rotation, we obtain a matrix that describes the permitted motions and forces in the e-e's frame as:

$$\boldsymbol{\Omega}(t) = \begin{bmatrix} \boldsymbol{S}_f^T(t)\boldsymbol{\Sigma}_f\boldsymbol{S}_f(t) & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{S}_\tau^T(t)\boldsymbol{\Sigma}_\tau\boldsymbol{S}_\tau(t) \end{bmatrix} \tag{5.24}$$

where $\boldsymbol{\Sigma}_\tau$ is a matrix like $\boldsymbol{\Sigma}_f$, but for the orientation components. Finally, we also define the dual matrix of $\boldsymbol{\Omega}$ as $\tilde{\boldsymbol{\Omega}} = \boldsymbol{I}_{m\times m} - \boldsymbol{\Omega}$. However, by construction, TEMPO already generates a sequence of end-effector poses aligned with the frame $\boldsymbol{\Sigma}_f$. Thus, the matrices $\boldsymbol{S}_f(t)$ and $\boldsymbol{S}_\tau(t)$ became identity matrix.

Thanks to these definition, we can define the two elements of $\boldsymbol{F}$, such that the control inputs belong to perpendicular spaces, $\boldsymbol{F}_m \perp \boldsymbol{F}_a$:

$$\begin{aligned}
\boldsymbol{F}_m &= \boldsymbol{\Lambda}(\boldsymbol{q})\boldsymbol{\Omega}\boldsymbol{F}_m^* + \boldsymbol{\Omega}\boldsymbol{F}_{null}, \\
\boldsymbol{F}_a &= \tilde{\boldsymbol{\Omega}}\boldsymbol{F}_a^*, \\
\boldsymbol{F}_{ccg} &= \boldsymbol{\mu}_r(\dot{\boldsymbol{q}}, \boldsymbol{q}) + \boldsymbol{p}_r(\boldsymbol{q})
\end{aligned} \tag{5.25}$$

where $\boldsymbol{F}_{null}$ is a term to set lower priority tasks in the null-space of the $\boldsymbol{F}_m$, $\boldsymbol{F}_{ccg}$ compensates the centrifugal, Coriolis, and gravity forces of the robot's dynamics, and $\boldsymbol{F}_m^*$ and $\boldsymbol{F}_a^*$ are the real unconstrained inputs to the controller.

Although the described controller efficiently separates force and motion to track a reference input signal, *e.g.* the resizing problem does not demand perfect tracing of the reference motion path. On the contrary, the robot must leverage the improved manipulability to obtain a path with a higher probability of success no matter the unexpected scenario. Thereby, the control design must reflect the necessity to comply with the environment.

5.3.2 Impedance Control

Given a dynamic system as per eq. (5.22), we select a control law for the input $\boldsymbol{F}$ with the following structure:

$$\boldsymbol{F} = \boldsymbol{\Lambda}(\boldsymbol{q})\boldsymbol{y} + \boldsymbol{F}_{ccg} \tag{5.26}$$

where $\boldsymbol{F}_{ccg}$, as per eq. (5.25), cancels the dynamics of the actual system, and $\boldsymbol{y}$ is a control term which addresses the desired behaviour of the system in the close-loop form.

The set of possible choices for the input $\boldsymbol{y}$, which is capable of reshaping the dynam-

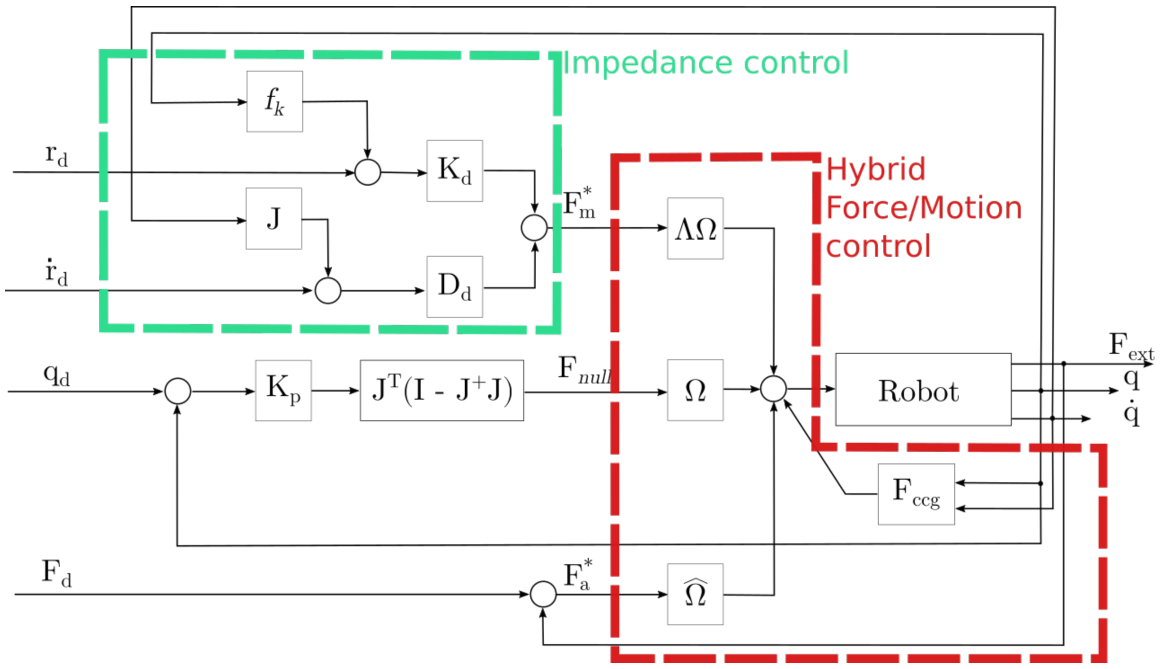


Figure 5.3: This scheme depicts the control structure for the proposed compliant hybrid force motion controller. The green dashed box highlights the impedance part for the motion space, F^*_m, and the red line marks the canonical hybrid force / motion controller.

ics of the system, composes the family of inverse dynamics controllers. A well-known approach in this set is described by the *Impedance Control* [Hog84], which takes into account all work exchanges between robot and environment, and every displacement of the robot's e-e behaves like a second-order system throughout the interaction, resulting in:

$$\boldsymbol{\Lambda}_d\ddot{\boldsymbol{e}} + \boldsymbol{D}_d\dot{\boldsymbol{e}} + \boldsymbol{K}_d\boldsymbol{e} = \boldsymbol{F}_d \tag{5.27}$$

where $\boldsymbol{\Lambda}_d$, $\boldsymbol{D}_d$, and $\boldsymbol{K}_d$ are positive definite (p.d.) matrices that select the desired behaviour of the interaction, with $\boldsymbol{\Lambda}_d$ the desired inertia, $\boldsymbol{D}_d$ the desired damping, and $\boldsymbol{K}_d$ the desired stiffness; $\boldsymbol{e} = \boldsymbol{r} - \boldsymbol{r}_d$ is the error between the current e-e position and the reference, and $\dot{\boldsymbol{e}}$ and $\ddot{\boldsymbol{e}}$ are the first and second derivative of $\boldsymbol{e}$; and $\boldsymbol{F}_d$ is the desired force, independent from the robot's configuration. Hogan [Hog87] showed how the close-loop controller holds its stability independently from the environment where the robot operates.

To yield the system in eq. (5.27), we select $\boldsymbol{y}$ as following:

$$\boldsymbol{y} = \boldsymbol{\Lambda}(\boldsymbol{q})^{-1}[\boldsymbol{\Lambda}_d\ddot{\boldsymbol{e}} + \boldsymbol{D}_d\dot{\boldsymbol{e}} + \boldsymbol{K}_d\boldsymbol{e} + \boldsymbol{\Lambda}(\boldsymbol{q})\ddot{\boldsymbol{r}}] - \boldsymbol{F}_d \tag{5.28}$$

where we care to notice that $\boldsymbol{\Lambda}(\boldsymbol{q})\ddot{\boldsymbol{r}}$ are external forces measured from the environment.

5.3.3 Proposed controller

We design a controller to perform the cutting of a surface with a knife using the Hybrid Force/Motion Control framework, but we propose to allow departures from the reference path to exploit the improved manipulability given by our TEMPO planner.

Although we maintain the original control law in (5.25), we design inner controllers for $\boldsymbol{F}_a^*, \boldsymbol{F}_m^*, \boldsymbol{F}_{null}$ to deploy specific behaviours, such as:

$$\begin{aligned} \boldsymbol{F}_a^* &= \boldsymbol{K}_a(\boldsymbol{F}_{ext} - \boldsymbol{F}_d), \\ \boldsymbol{F}_m^* &= \ddot{\boldsymbol{r}} + \boldsymbol{\Lambda}(\boldsymbol{q})^{-1}[\boldsymbol{D}_d\dot{\boldsymbol{e}} + \boldsymbol{K}_d\boldsymbol{e}] \\ \boldsymbol{F}_{null} &= \boldsymbol{J}^T(\boldsymbol{I} - \boldsymbol{J}^\dagger\boldsymbol{J})(\boldsymbol{K}_p(\boldsymbol{q} - \boldsymbol{q}_d)) \end{aligned} \tag{5.29}$$

The action force is a simple tracking of the desired force $\boldsymbol{F}_d$ with $\boldsymbol{K}_a$ weighing the performance. To counteract the noisiness of the measurements on the external force ($\boldsymbol{F}_{ext}$), we implement a moving window filter with the size of 300 samples.

As per the motion force, we propose to endow the controller with an impedance behaviour described by the matrices $\boldsymbol{D}_d$ and $\boldsymbol{K}_d$. Because we do not need to have a full dynamics replacement, we decide to have the inertia matrix of the system the same, which yields the system being passive. Finally, we ask the controller to achieve a secondary task in the *null* space of the motion. Although the tracking of the trajectory is not mandatory, we ask the robot to minimise the distance between the current configuration, $\boldsymbol{q}$, and the desired one, $\boldsymbol{q}_d$. This task at lower priority preserves a motion that is closer to the optimal $\boldsymbol{\xi}^*$ and, hence, the manipulability properties.

5.4 Experiments

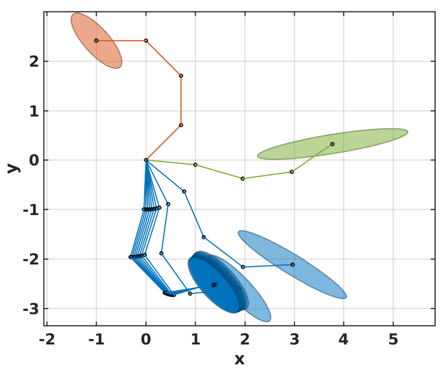
We run a collection of experiments to assess our method in either simulated or real scenarios. Firstly, we measure our objective's performance in reaching a target manipulability ellipsoid and whether it is suitable for tracking feasible and unfeasible ellipsoids. The *point-wise* section works out a set of experiments on a planar robot composed of four revolute joints with the z-axis point outward the x-y plane. Therefore, the entire robot's movements lay on the x-y plane and within a 4[m] radius from the origin; each link has a unitary length. Because we are working in a planar environment, the manipulability ellipsoid collapses to an ellipse, which is easier to report alongside the robot's movements.

We design a regulator which rearranges the robotic joints to minimise the distance between current and target manipulability ellipses. We perform two experiments investigating the regulator's performance with feasible and unfeasible ellipses. While the former account for the robot kinematics, the latter are kinematically agnostic and designed in Operational space.

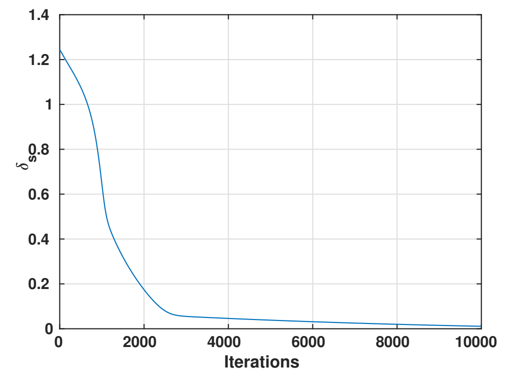
Secondly, we test the performance of TEMPO *motion-wise*. We generate a trajectory for the four joints robot and ask TEMPO to minimise the objective cost of eq. (5.2). The objective accounts for manipulability and smoothness. We run the experiment on the more challenging case of a kinematically agnostic ellipse.

Thirdly, we compound the previously tested behaviours with the end-effector constraint, Sec. 5.2.6. In this set of experiments, we use a seven dof Panda arm manufactured by Franka Emika, and we moved to a C++ implementation of our approach. We perform two sets of analyses to answer two questions: does TEMPO improve both smoothness and manipulability while constraining the end-effector on the PC?; and what is the beneficial effect of TEMPO over other methods? We compare TEMPO against RRT*-RMM to obtain a fair comparison in terms of manipulability awareness.

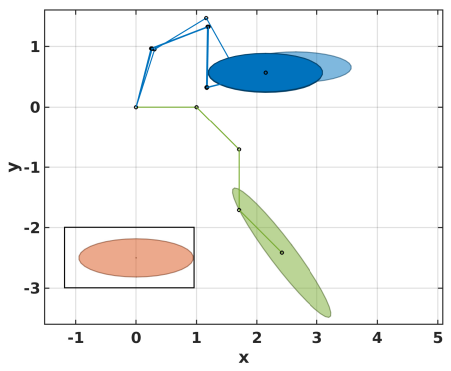
Finally, we test the proposed hybrid force/motion controller on the Panda. We report the algorithm's behaviour on different surfaces, *i.e.* flat plane, inclined plane, and curved surface, and we show that it safely complies with unexpected obstacles in the environment.



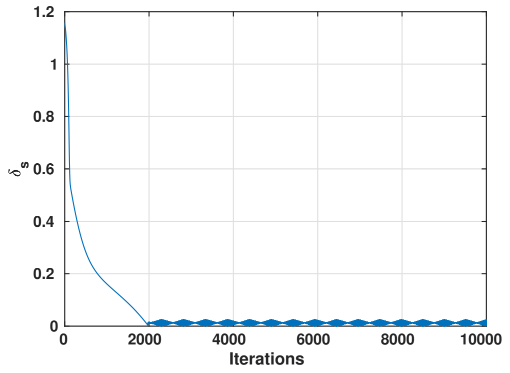
(a)



(b)



(c)



(d)

Figure 5.4: The first row shows the kinematic-aware experiment. The robot starts in a random configuration, coloured in green, and a target manipulability ellipse is obtained by a random configuration, coloured in red. The blue configurations show the iteration of the algorithm, which converges to the desired ellipse after 10k iterations. Fig. 5.4(b) shows the monotone descreasing objective value over iterations. The bottom row shows the non-kinematic-aware experiment. The robot starts in a random configuration, coloured in green, and a *task-informed* manipulability ellipse is selected using the task information, coloured in red in the picture-in-picture. The blue configurations show the iteration of the algorithm, which converges to the desired ellipse after 10k iterations. Fig. 5.4(d) shows the monotone decreasing objective value over iterations.

5.4.1 Point-wise gradient validation

The core of TEMPO minimises the objective function in eq. (5.8), which accounts for the distance between the current manipulability matrix and the *task-informed* matrix using the gradient of the S-Divergence.

To assess the accuracy of the computed gradient, we design a set of experiments on a 4 revolute joints manipulator lying on the x-y plane. Although the simpler structure constraints the robot to move only in the 2D space, it makes it easier to visualise how the gradient affects the ellipsoid, which results in an ellipse for this case.

A steep descent algorithm is implemented using the momentum strategy [Qia99] to increase the convergence rate and value past descending directions more.

Two sets of experiments test the performance of the gradient under different inputs: a random starting configuration and the desired ellipse. Firstly, we feed the algorithm with a reachable manipulability ellipse, which is generated by taking a random robotic configuration and computing the actual manipulability matrix. Secondly, we shape an ellipse without any knowledge of the robot kinematic, and we give it as input to the algorithm.

Both the experiments task the algorithm to find a configuration matching the given ellipse, but no constraints on the position or orientation are made. Manipulability ellipses are not unique for all configurations, and two entirely distinct joints' values may yield the same matrix, *e.g.* for the considered 4 dof robot, the manipulability ellipse $\boldsymbol{M}$ obtained by the configuration $\boldsymbol{q}$ is the same as the one computed flipping the first joint of π radiant. Therefore, generally, we do not expect to obtain the same configuration used to compute the input matrix. An Inverse Kinematic algorithm (IK) that accounts for the manipulability matrix is proposed by [JRCC0]. In their work, the authors use the learning by demonstration to generate new possible configurations for the robot with a specific manipulability matrix. However, this approach is prone to the evaluation of a considerable number of demonstrations, and it is suitable only for reachable manipulability matrices. In our work, we aim to release the former requirements and investigate the behaviour of a mathematically sound approach under either kinematic-aware or non-kinematic-aware ellipses.

In Fig. 5.4, we report the result of the experiments. In the first row, the algorithm must search for a robot configuration that matches an ellipse given as input. The desired ellipse is selected by computing the manipulability matrix of a randomly picked configuration for the 4 dof robot, depicted in red in Fig. 5.4(a). Then, an initial random configuration (showed in green) is selected, and the algorithm starts to optimise similarities between the ellipses. After 8000 iterations the algorithm converges to a ellipse (in dark blue) with a computed distance $\delta_s^2 = 0.001$. In Fig. 5.4(b) the convergence rate of the algorithm over the 10k iterations is reported; the algorithm greatly decreases the objective function in the first 3000 iterations, and, then, it reaches a plateau that monotonically minimises the metric and obtains its minimum at the end of the iterations, with a δ_s lower than $1.04e^{-4}$. We may notice that TEMPO converges to the right ellipse but not the original configuration; this is because no unique mapping between manipulability ellipses and joint space exists, as we discussed before.

The bottom row of Fig. 5.4 reports the result for the experiment with a *task-informed* ellipse, which is designed using consideration of the forces involved by the task in Cartesian space. The designing stage of the desired ellipse does not account for the size or limitations of the robot, which may result in a matrix that is unfeasible for the kinematics at hand. The initial conditions of the algorithm are the same as the former example, with the initial configuration picked random. As we expect, the algorithm converges to a configuration that minimises the distance between the current manipulability ellipse and the desired one after less the 2000 iterations. Then, the S-Divergence objective starts oscillating around the mean value of 0.0014 with variance ± 0.001 for the left 8000 iterations. In the picture-in-picture of

Fig.5.4(c), we show the *task-informed* ellipse selected for the experiment. Because of the unawareness of the robot kinematics, the chosen desired matrix might be unreachable from the robot, and the algorithm reaches the best solution at minimum δ_s. However, two configurations close to one another may yield equal values. In this case, the momentum strategy sees the two configurations as local minima, and it produces swings to escape, with the results of oscillations between the two robot configurations.

The definition of the *task-informed* manipulability ellipse in Cartesian space has two beneficial effects:

1. it is more natural for the operator to select an ellipse in Cartesian space rather than

using the joint space

2. it allows to use of a robot-agnostic model of the task

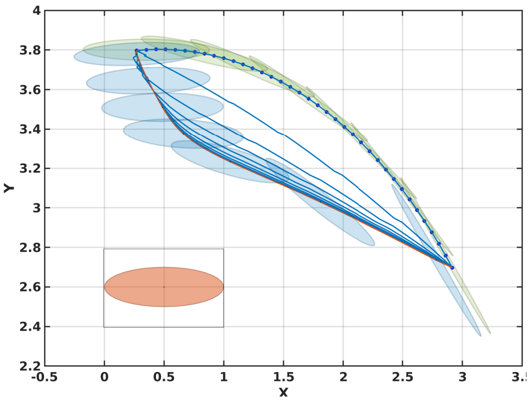
The downside of our approach lies in the convexity behaviour of the S-Divergence, which guarantees to be convex only within a certain range, $\boldsymbol{A} \geq (1+\sqrt{2})\boldsymbol{B}$. Therefore, if the commanded ellipse spans farther out than the available range, the metric will become concave, and the algorithm might fall into local minima.

5.4.2 Motion-Wise - Planar TEMPO

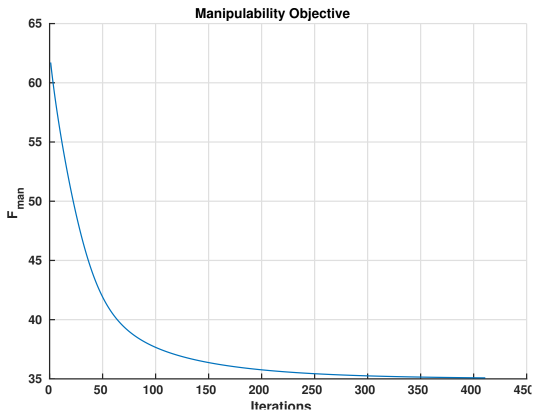
Now that we have validated the gradient of the S-Divergence, we use our TEMPO algorithm to optimise an entire path for the 4 dof robot. In this experiment, we do not constrain the robot e-e to stay in touch with any surface, and it can freely move in the 2D plane.

We provide the algorithm with an initial path and the desired *task-informed* ellipse $\hat{\boldsymbol{M}} = [0.99, 0; 0, 0.1]$ whose major axis is on the x-axis and minor axis on the y-axis (red ellipse in the bottom left of Fig. 5.5(a)). The starting and destination point of the path are (2.80, 2.68) and (0.26, 3.78) respectively, and the initial path draws a semicircular curve connecting the two points, depicted by the blue asterisks in Fig. 5.5(a); in the same picture, we also report the manipulability ellipses along the initial path in soft green colour. Thus, we task the algorithm to minimise the objective function in eq. (5.2), which accounts for both smoothness and manipulability over the trajectory.

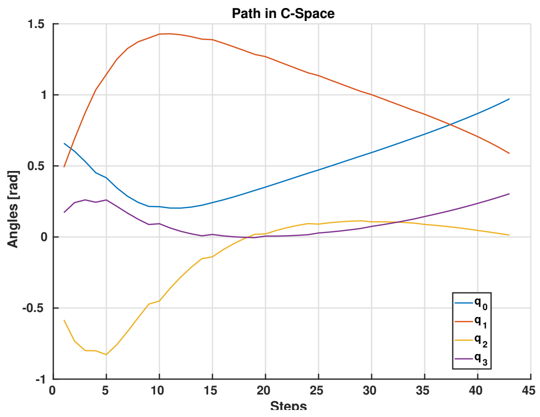
Fig. 5.5(a) shows the history of the paths proposed by TEMPO every 100 iterations (blue solid lines), and we finally show the manipulability ellipses of the optimal trajectory in soft blue. The tail of the optimised trajectory has the ellipses very close to the *task-informed* one, while the head of the path does not reach the desired shape. Although the behaviour may look unsatisfactory at the first glance, it is consistent with the requirements of smoothness and starting and destination points. Looking at the initial point of the path, we see the ellipse being tilted of almost 90 degrees with respected to desired one. Although



(a)



(b)



(c)

Figure 5.5: Fig. 5.5(a) shows the iterations of the TEMPO algorithm. The initial path is in green and the target ellipse is coloured in red. The blue lines show the robot trajectory over iterations, and we show the manipulability ellipse in blue for the optimised trajectory. The manipulability objective smoothly decreases as shown in Fig. 5.5(b). The last picture shows the resulting trajectory in joint space (C-Space).

the algorithm drives away the path from this unwanted ellipse, it cannot change the initial position, therefore, it consistently adjusts the following points to obtain the best smooth result.

At the beginning of the optimisation, the value of the manipulability objective, $\mathcal{F}_{man}$ is 61.71 and drops monotonically to 37.09 after 407 iterations. Because we account for the smoothness along the path and selected an ellipse unawarely of the robot kinematic, the manipulability objective does not reach zero. However, the proposed trajectory has improved the manipulability to reach the most accurate representation of the *task-informed* ellipse, with an improvement of 40%. In Fig. 5.5(a), we report the convergence rate of

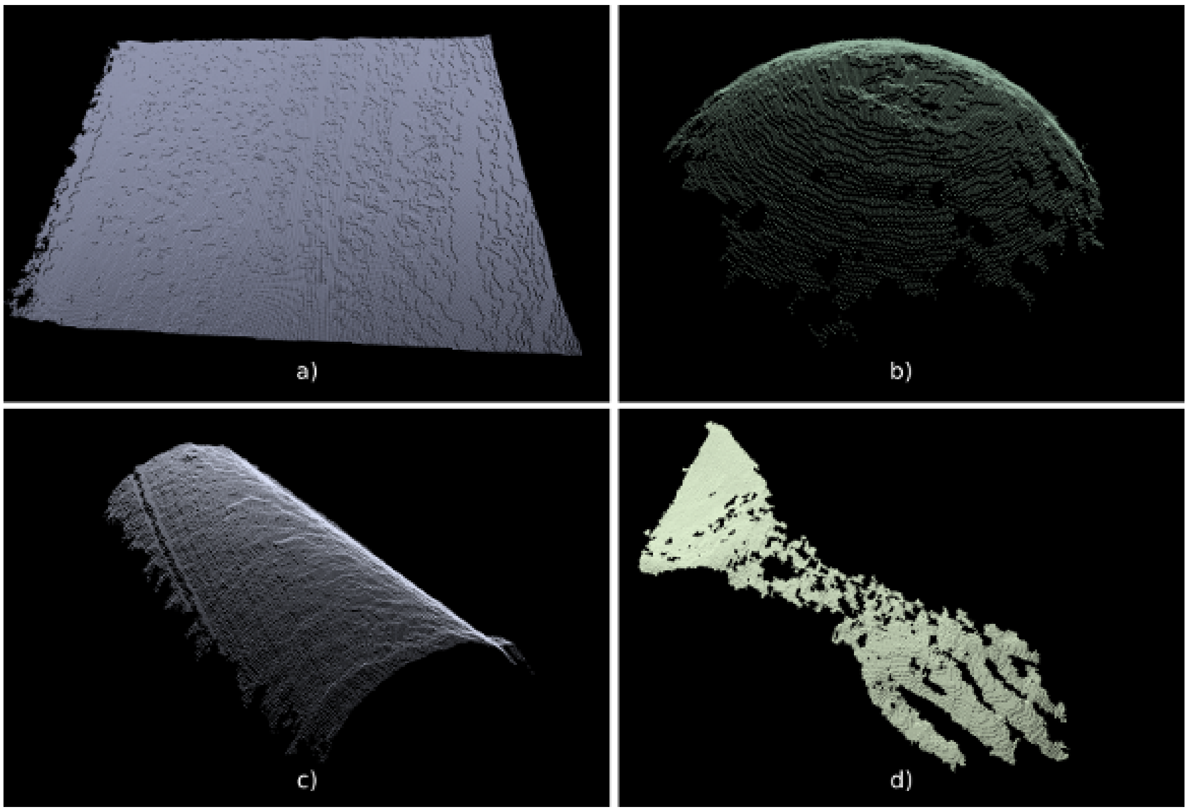


Figure 5.6: The figure shows four PCs of different objects used in the experimental section. Fig. 5.6a) is the flat plane of a table, which is posed in front of the robot base. Fig. 5.6b) is a bowl posed at 0.6 m from the robot's base. Fig. 5.6c) shows a pipe of 20 cm diameter located diagonal to the robot's base. Fig. 5.6d) reports one glove, which is a typical waste for the nuclear industry.

the algorithm, and, in Fig. 5.5(c), we show the proposed trajectory in configuration space for the 4 dof robot.

The value of λ, in eq. (5.2), plays a critical role in the behaviour of the algorithm. High values of λ result in smoother paths that will converge to a straight line when the manipulability term is negligible. On the other hand, low values of λ cause the path to be jerky, especially at the first and last step of the paths, and it will converge to a path composed of only three configurations: the fixed initial and destination, and a middle one that yields the best manipulability ellipse. After a few experiments, we set λ to 0.1, which shows a good trade-off between the two objectives.

5.4.3 Motion-Wise - TEMPO

We test our algorithm, TEMPO, on a set of PCs taken with a low-cost sensor, the Astra Orbbec camera. Because of the depth sensor quality, the camera provides the algorithm with noisy PCs that represents the surface onto we want to plan a path for the e-e.

We select 4 objects that are common wastes in the decommissioning process of the nuclear industry, a flat plane, a pipe, a bowl, and a glove. The collection of surfaces covers a broad range of curvatures and complexities, from the simple flat plane with no curvature to the more complex glove, where several curvatures are present. Moreover, the low accuracy of the camera left holes and stripes of missing data over all the point clouds. In Fig. 5.6, we show the point cloud of each object.

In Fig. 5.7, we report 20 paths generated on the four point clouds. 5 examples per each pointcloud. We select the initial and destination points on the surface of the object by using points already in the point loud. A straight line between the obtained points is fed to TEMPO, which optimises the trajectory based on the two objectives aforementioned.

These trajectories depart from the shortest path but comply with the manipulability requirement. The images show that the algorithm generates either short or long paths on the surfaces, and it successfully obtains a solution with high curvatures, like in the bowl and glove case.

We break down the optimisation process by saving the trajectory at several optimisation stages. Fig. 5.8 shows the optimisation procedure over iterations. It starts with the initial path in red, generated using the RRT*-RMM algorithm. Then, in blue we show the proposed path every 1000 iterations and the final trajectory is coloured in magenta. We may notice how the optimisation greatly improves the smoothness of the overall motion, which numerically results in a 35.4% improvement on the relative objective. Moreover, the manipulability improves as well with $\mathcal{F}_{man}$ values reducing to 9.0237%.

Although the objective functional $\mathcal{U}$ reaches its minimum at zero, the presence of constraints in the optimisation increases this value. Throughout the procedure, we are considering two constraints: one, explicitly defined, is the force that the robot's e-e must

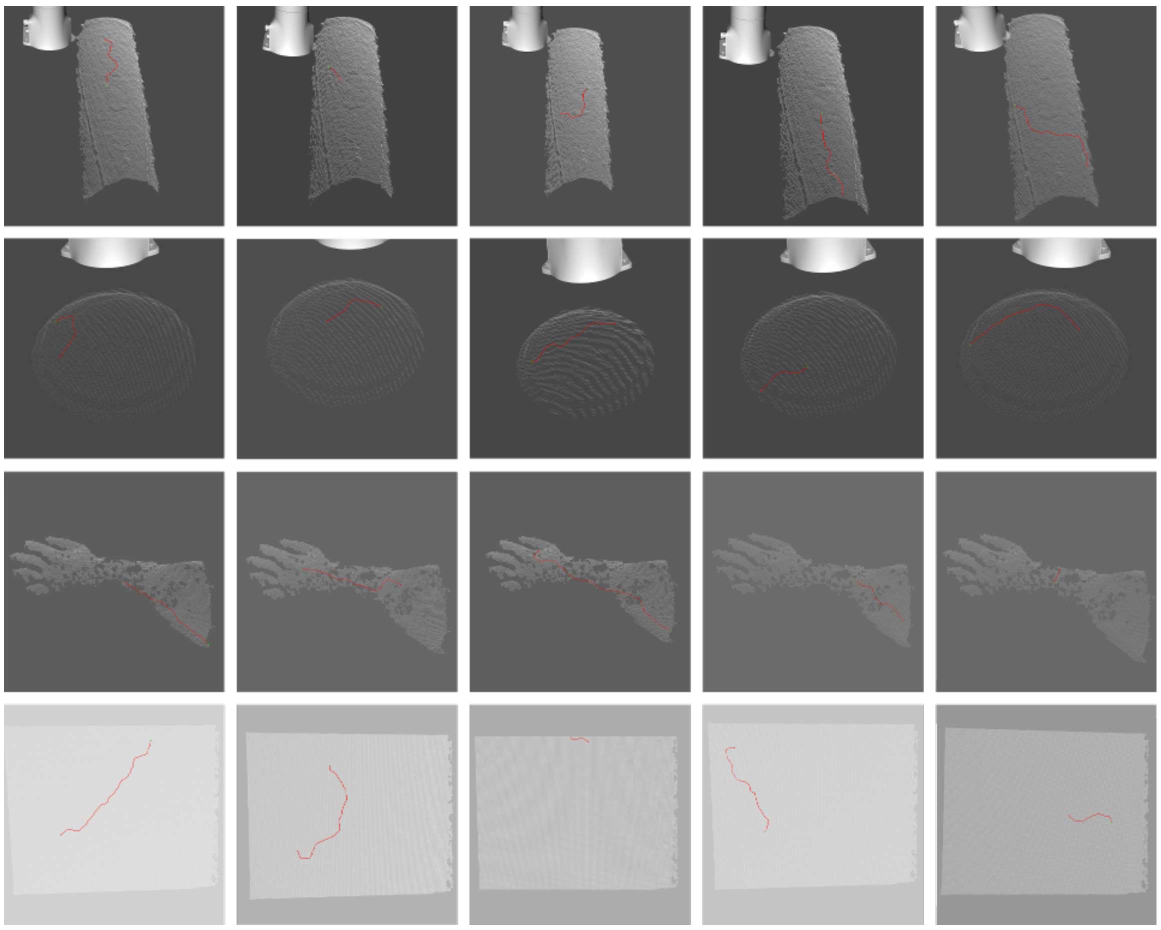


Figure 5.7: This picture shows the result of the TEMPO algorithm on four PCs: a cylindrical barrel, a bowl, a glove, and a flat plane. Initial and destination points are selected randomly from the pointcloud set. The algorithm starts with a linear path connecting two points and optimises the waypoints accordingly to the objectives.

exert on the surface at each time step; the other is implicit by the desired manipulability ellipsoids, which might not match any ellipsoid producible by the robot's kinematics.

The path proposed by RRT*-RMM is slightly off the surface of the barrel, and our approach successfully managed to reduce this initial displacement and obtain a path clinging to the PC of the object. On the other hand, we may notice how the starting point of the trajectory stays off the Point cloud. This behaviour complies with one of the constraints, which requires keeping the starting and destination points fixed.

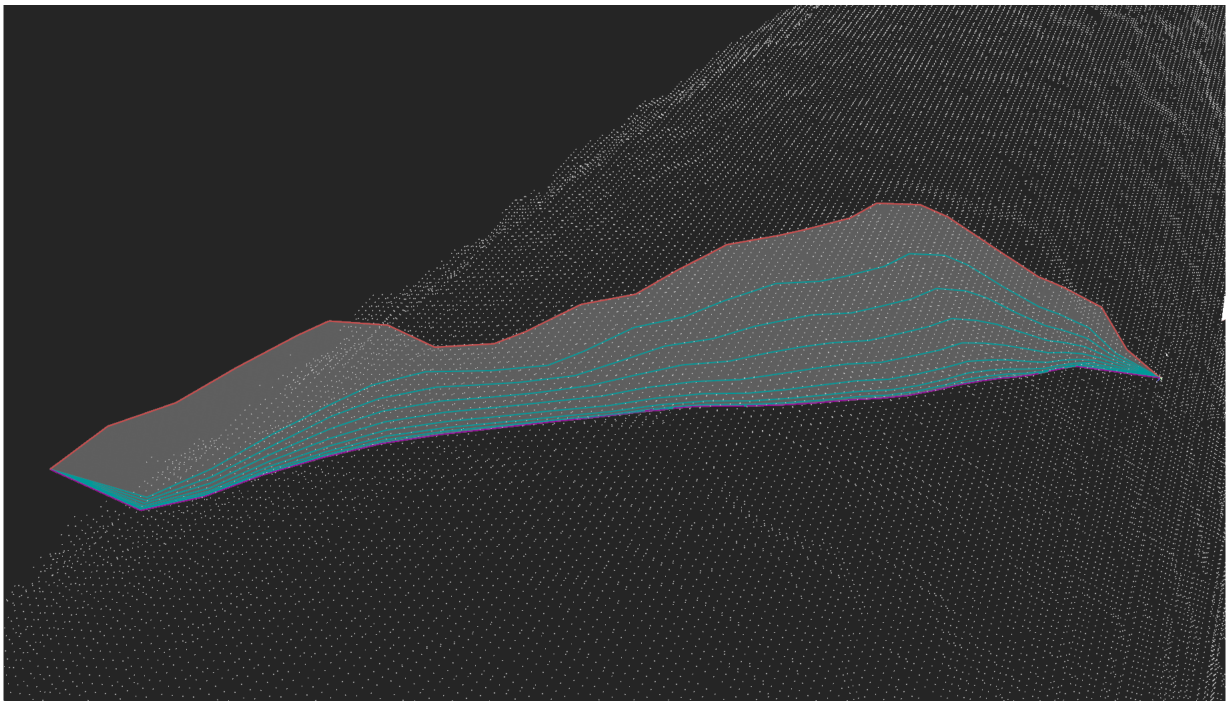


Figure 5.8: TEMPO trajectory on a barrel, initial path in red with U equal to 267.964, $\mathcal{F}_{smooth} = 81.6983$ and $\mathcal{F}_{man} = 39.2141$. The algorithm converges to a new trajectory (in magenta) after 8756 iterations with new values: U equal to $F_{smooth} = 52.727$ and $\mathcal{F}_{man} = 38.2821$

5.4.4 TEMPO vs RRT*-RMM: a comparison

The presented algorithms, RRT*-RMM and TEMPO, share the common idea of improving the manipulability throughout the path. However, they build onto two radically different structures. RRT*-RMM is a sample-based approach, while TEMPO is an optimisation based approach. Also, the former accounts for manipulability altogether, while the latter considers the shape of the manipulability matrix.

Because of the different approaches, we propose to assess the results of the two algorithms in a typical scenario. Both algorithms need to provide a trajectory for the robot (a Franka Panda arm) onto the surface of an arbitrary object, which is sensed through a noisy PC.

The object has a fixed position and orientation throughout all the experiments, and the observed surface is fully reachable by the robot kinematics. As discussed in the previous chapter, initial and destination points affect the algorithm's performance. Thereby, we

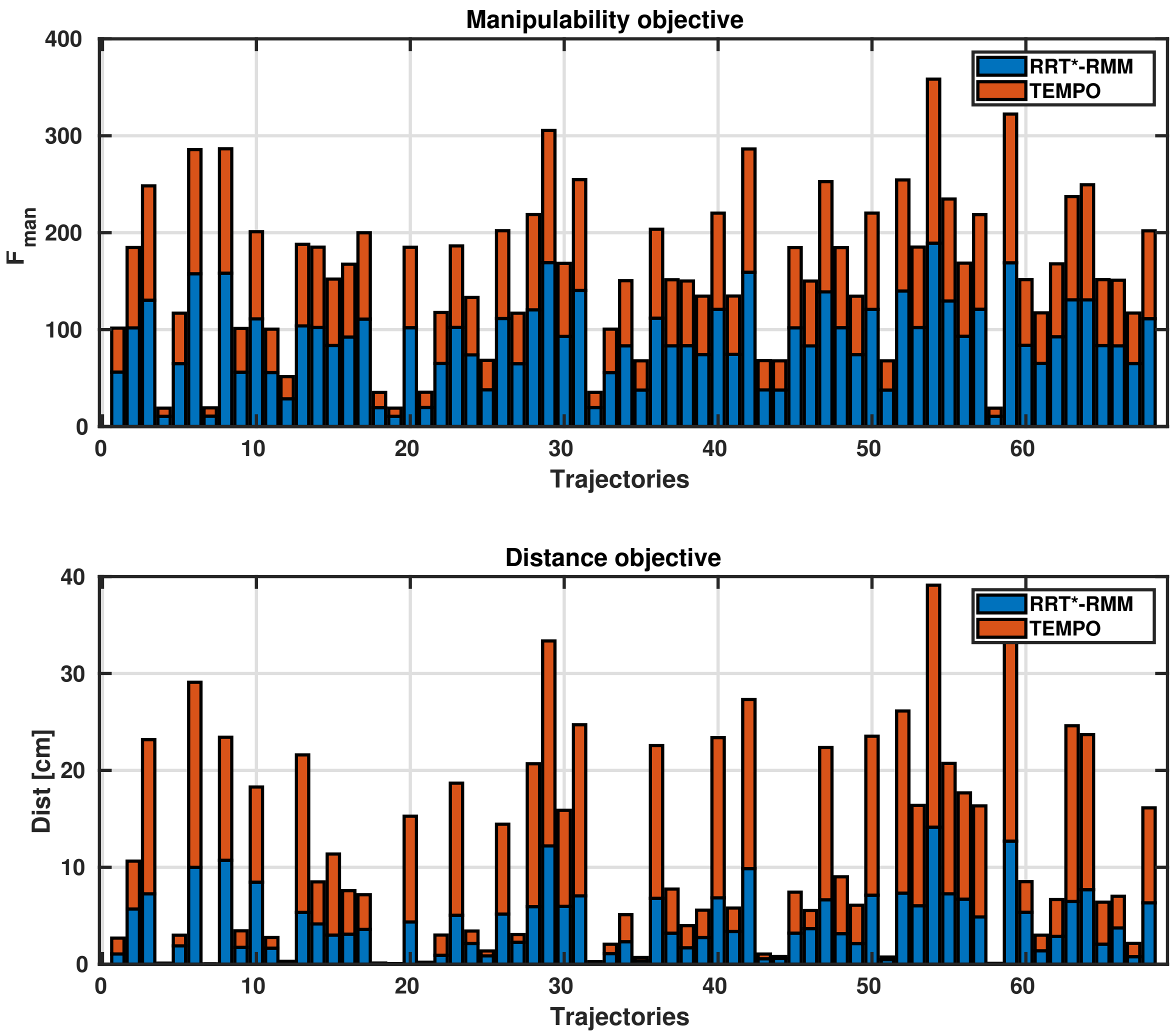


Figure 5.9: This picture reports the comparison between RRT*-RMM, in blue, and TEMPO, in red. The upper bar graph shows the manipulability value, F_{man}, over 68 different trajectories. The bottom bar graph accounts for the distance covered by the proposed paths. TEMPO yields an average improvement of 18.5748% compared to RRT*-RMM in terms of manipulability, but it produces longer paths with a +44.026% on the path length.

randomly sample two points from the PC and set them as the initial and destination points; as for the orientation, we pick the inward normal to the surface at the point of interest. This procedure reduces the risk of unfavourable initial configuration due to the inverse kinematic solver. Once a pair of points is obtained, we run both RRT*-RMM and TEMPO using the same pair. We analysed a total of 68 trajectories.

TEMPO needs the *task-informed* manipulability ellipsoid, $\hat{\boldsymbol{M}}$ as input. In this experiment, we select:

$$\hat{\boldsymbol{M}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.01 \end{bmatrix}$$

TEMPO optimises each path until it reaches either the maximum number of iterations (set to 200k) or convergence (the step size is below $1e^{-5}$).

Similarly, RRT*-RMM runs for a minimum of 10k iterations and returns the first feasible path. The minimum number of iterations reduces the possibility of serendipity, in which the algorithm returns after a few iterations and not enough exploration is performed.

In Fig. 5.9, we report the value of the manipulability objective and the distance covered by the proposed paths. In red, we show the TEMPO results and in blue RRT*-RMM results.

As expected, TEMPO performs better in terms of manipulability objective, with paths that are closer to the *task-informed* manipulability matrix, (5.4.4), of about 18.5748% on average and with a standard deviation of 2.467. On the other hand, TEMPO takes longer departures from the shorter path, which led to an increment of slightly more than 44% on the path length. Short paths result in a similar length between RRT*-RMM and TEMPO, but with a slight improvement in manipulability by TEMPO.

The 54th trajectory has the worst trade-off between manipulability and length. Although the manipulability objective sees an increment of almost 100%, TEMPO needs around three times the length computed by RRT*-RMM. Looking at this path, we notice that RRT*-RMM provide a jerky path. The TEMPO smoothness objective extremely discounts these types of movements. Hence, TEMPO had to take a long departure to comply with the smoothness requirement and align the tool orientation.

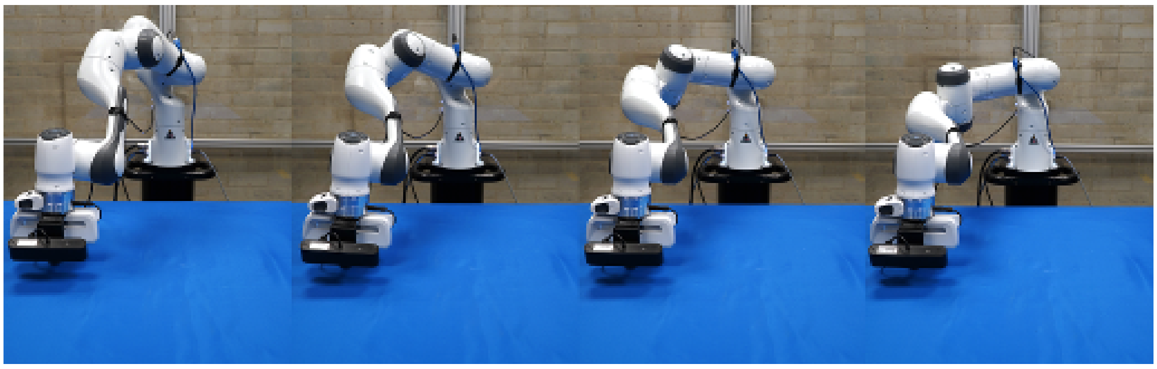


Figure 5.10: The robot is tasked with following a circle in front of the robot base while exerting a force on the table. Cancelling out the Null-space component of the controller, the robot bends the elbow. After three iterations (foremost right image), the robot is not capable to continue the trajectory without triggering an error on the joints' limit.

5.4.5 Compliant hybrid force/motion controller

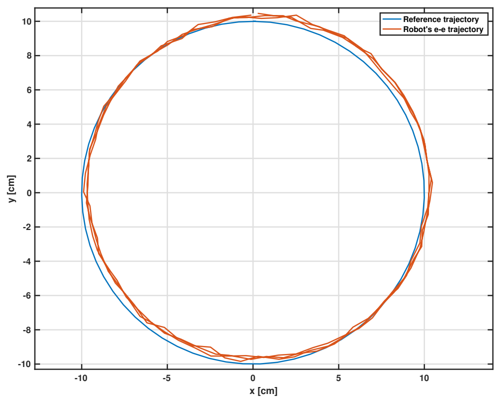
Generally, cutting tasks require the robot to carry out trajectories onto the x-y plane of the end-effector reference frame while exerting forces along the z-axis. Hybrid force/motion controllers cast movements and force applications in orthogonal spaces, and we may control each action independently.

The proposed controller, Sec. 5.3, permits the robot to show a compliant behaviour on the x-y plane and exert forces on the z-axis.

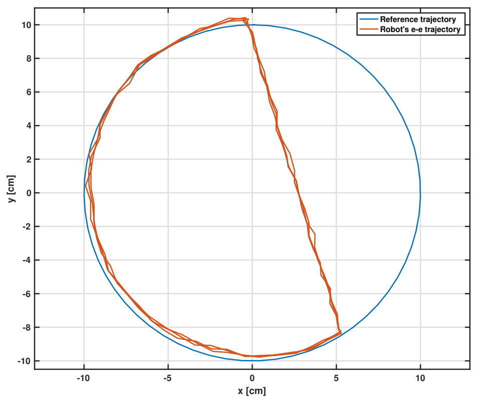
We test this behaviour on three incrementally complex surfaces: a flat plane, an inclined plane ($\tilde{2}0$ degrees), and a curved surface (a cylinder). In Fig. 5.12(a), the end-effector must follow a circular path on the plane. The robot stays in touch with the surface throughout all execution and applies the required force of 3 [N] perpendicular to the plane.

The inputs of our controller are in the end-effector space, and our method bridges those inputs to robot configurations. Therefore, we do not have direct control over the joint trajectory, which is adjusted indirectly.

Thanks to the *Null-space* component of our controller, the robot exploits the redundant degree of freedom to maintain the desired configuration. If we discount this part of the controller, the robot starts leaning over the left side until it reaches the limit stop due



(a)



(b)

Figure 5.11: The robot is tasked with following the blue reference trajectory, which is a circle of 10 [cm] radius. The red lines report three iterations of the end-effector movements. The impedance controller creates a displacement with the reference trajectory due to the unaccounted friction between end-effector and table. The error is always less than 0.3 [cm]. On the right side, we report the compliant trajectory because of the obstacle encumbering the robot's trajectory.

to the joint's limits. Fig. 5.10 shows this issue on four laps of the circle. In the leftmost picture, the robot keeps the original configuration. However, at every lap joints number 0, 1, 4, and 5 move to keep the right orientation at the end-effector. This problem has a twofold negative effect: it brings the robot to configurations that might make it impossible to continue the trajectory and changes the manipulability values from the original trajectory.

Because of the added contribution, the robot maintains the desired configuration, hence, the manipulability value provided by TEMPO.

As for the first experiment, we report the error tracking of the robot's end-effector in Fig. 5.11(a). The error persistently remains under 0.3 [cm], but it never reaches zero. In this test, the controller does not have any information on the table's friction. Thereby, the system assumes the unaccounted term as an external pull and shows a compliant behaviour against it. On the other hand, the force is 3 [N] over the entire trajectory with a maximum error of 0.213 [N]. The compliancy of the motion space is even more evident when an obstacle encumbers its way, Fig. 5.11(b) and Fig. 5.12. Here, the robot

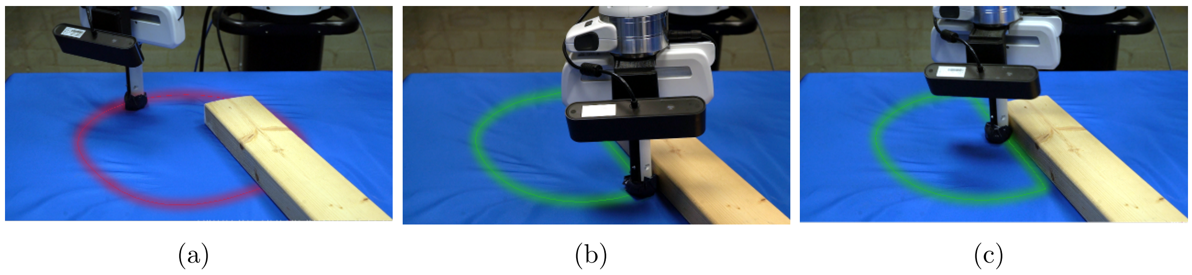


Figure 5.12: In the Fig. 5.12(a), we display the reference trajectory for the robot with a red colour. An obstacle blocks the right side of the trajectory. Once the robot makes a contact with the obstacle, the impedance controller allows the end-effector to follow the obstacle's shape until the original path may be resumed, green path in Fig. 5.12(b) and 5.12(c)

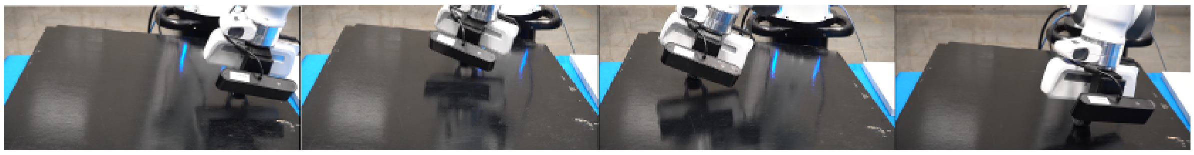


Figure 5.13: The robot is tasked with following a circle on a 20 degrees inclined plane while exerting a force on the table. The robot is able to follow the circle and keep the desired orientation throughout the entire movement.

is incapable of tracking the desired position and edges the unknown shape until it may resume its commanded trajectory.

In the second experiment, the robot executes the same circular path, but with the head tilted at 20 degrees from the plane. The controller accounts for the orientation and commands the robot accordingly. In Fig. 5.13, we see a sequence of actions of the robot executing this task. Although this test increases the complexity by introducing an orientation, the performance is comparable to the flat plane example, with mean and standard deviation close to the first experiment.

The third experiment further increases the complexity of the trajectory. It addresses the robot behaviour while following a path provided by TEMPO onto a cylindrical barrel. We assume the barrel is either fixed on the ground or heavy enough to avoid slippage once the robot begins interacting with it. In this case, we bolted the cylinder (made of thick cardboard) to a steel plate (around 2 $[kg]$ weight). Thus, we can assure the object stays still throughout the entire task.

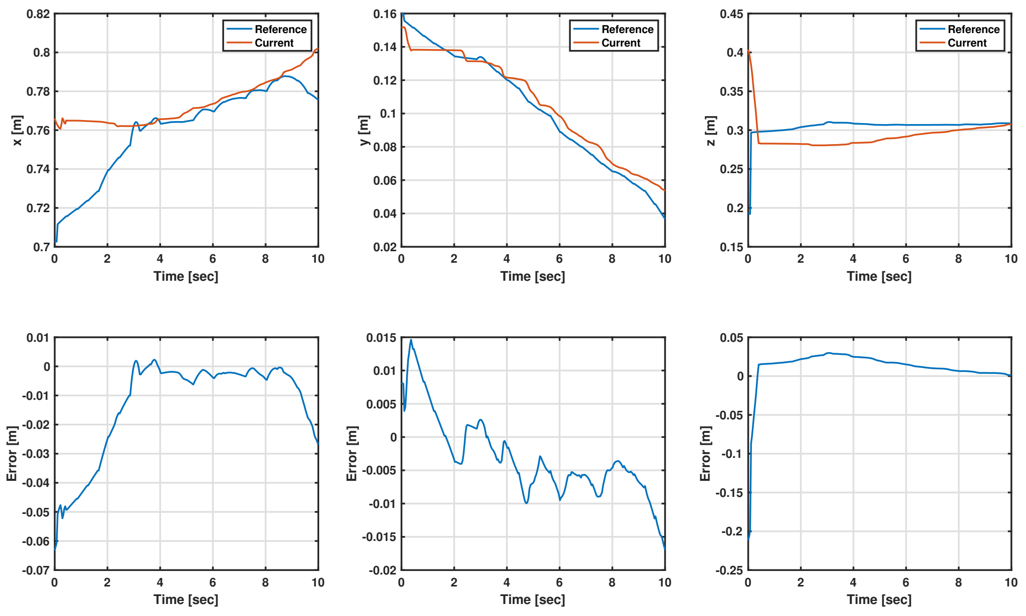


Figure 5.14: This picture reports the errors on the $[x, y, z]$ of the end-effector against the reference trajectory.

The trajectory proposed by TEMPO makes a curved line on the barrel surface, starting from the right side (facing the base of the robot) and ending at the mid-left of the barrel shape, Fig. 5.15. The end-effector keeps the z-axis always oriented as the inwards normal of the touched surface, and the path has the x vector toward the next point.

Fig. 5.14 reports the robot positional errors on the x-, y-, and z-axis. The blue line on the top raw is the reference end-effector trajectory, while the red line represents the current robot position. The best performance are reached on the y-axis with a mean error of $\mu_y = -0.008[m]$ and a standard deviation of $\sigma_y = 0.005[m]$. On the other end, the z-axis accumulates the most error with a mean error of $\mu_z = 0.0255[m]$ and a standard deviation of $\sigma_z = 0.0259[m]$. The x-axis performs slightly better than the axis z, but with two large discrepancies at the initial and destination of the path, $\mu_x = -0.02[m], \sigma_x = 0.01668[m]$.

We believe that the latter phenomena is due to some error in the camera calibration. Because the x-axis is oriented from the base of the robot onward, and the trajectory follows the same profile, a little offset in the camera calibration may force the robot to push onto the object creating the larger error.

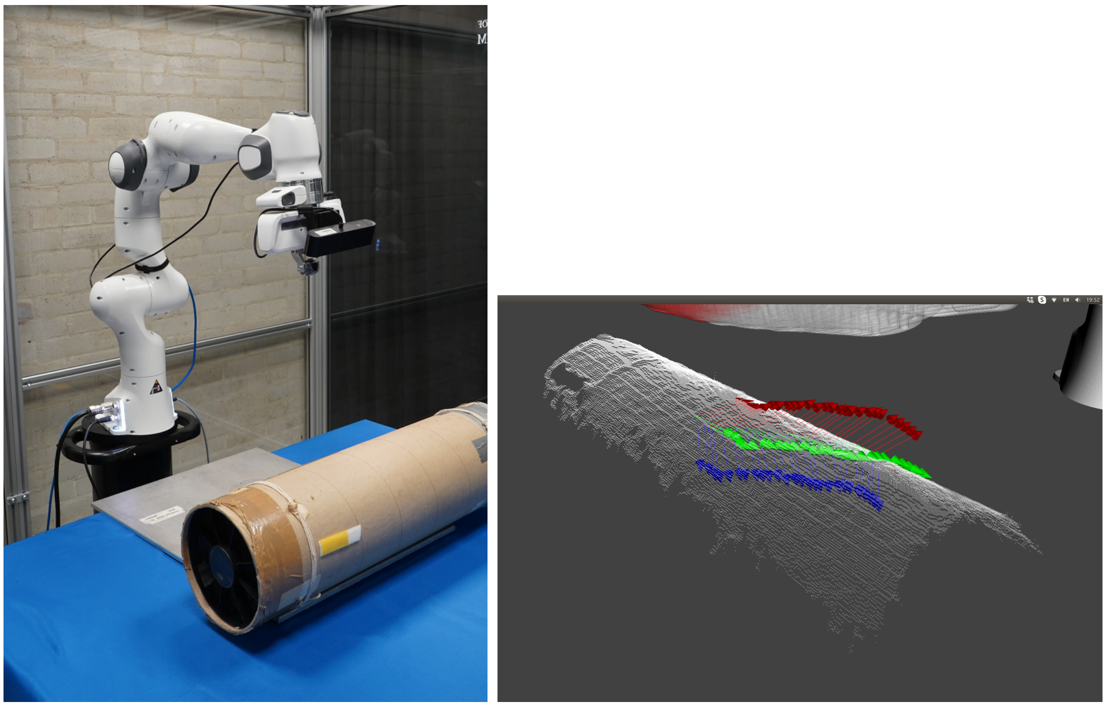

(a) (b)

Figure 5.15: The robot setup for the experiment 3 is Fig. 5.15(a). A cylindrical object is posed at 0.5 [cm] in front of the robot base. We scan the object and enquire TEMPO for obtaining a trajectory, Fig. 5.15(b). The trajectory is onto the surface, and with normals inward to the object's surface.

As for the forces, we decided to select a small gain for the force controller and allow some distance between the commanded and the current force. This choice reduces the jumps on the z-axis if the robot loses contact with the surface. Therefore, we do not obtain a perfect tracking of the desired force. Fig. 5.16 shows the tracking results of the force. The measured force stays close to the commanded profile but with displacements up to $0.75[N]$ (on average $0.25[N]$).

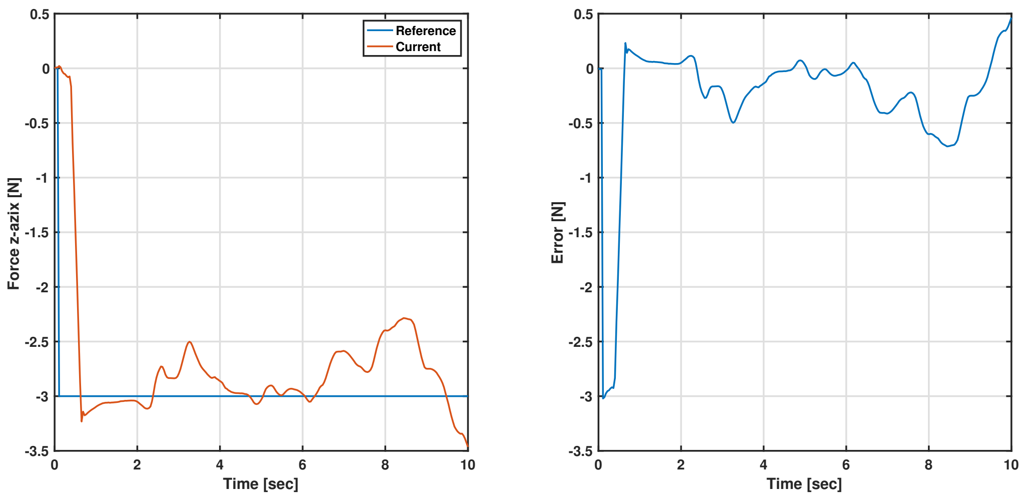


Figure 5.16: We set a constant force profile of 3 [N] as input, blue line. We select low gains for avoiding unexpected jumps of the end-effector, if we lose contact between object and end-effector.

5.5 Conclusion

This chapter addresses the planning of trajectories for robots whose end-effector stays in contact with an arbitrary surface. The proposed path generator belongs to the optimisation-based planner class and accounts for the smoothness of the path and the shape of the manipulability ellipsoid throughout the task execution. We introduced a novel objective functional, which balances two terms: smoothness and manipulability. The latter component relies on the S-Divergence metric, which allows considering the full manipulability matrix instead of an approximated scalar. We show that our approach is robust to track either feasible and unfeasible matrices for the robot and that for the latter case, our method yields the closest possible solution. This point grants kinematically-naive users to design manipulability ellipsoid solely based on their expertise in the task.

To exploit the improved manipulability, we propose a compliant hybrid force/motion controller. This methodology splits the motion and forces application and permits to jointly follow the desired path and exert a force in the normal direction.

Firstly, a set of experiments test the efficacy of the proposed planner on a 2D scenario both in the point-wise tracking and the motion-wise optimisation. Our planner minimises the objective function, and we show the breakdown of the smoothness and the manipulability contributions. Also, our experiments show that our planner tracks the manipulability matrix that may not be feasible by the robot and reaches the closest configuration. Secondly, we perform a number of 3D experiments with the Panda robot manufactured by Franka Emika and using noisy point clouds extracted by an Astra Orbbec camera. Here, we show that TEMPO yields optimum trajectories on arbitrary surfaces and constrains the robot e-e to always touching the surface. Furthermore, we propose a comparison between TEMPO and RRT*-RMM that sheds light on the improvements of the former over the latter. Finally, we test the hybrid force/motion controller on three surfaces at greater complexity. The controller allows the tracking of the desired path on all experiments. We quantitatively report the tracking errors of the controller.

CHAPTER 6

Task-informed grasping

6.1 Introduction

Consider the task of unloading a box full of books and placing them in a library. First of all, we pick every book one by one. Then, we place it in the right location beside other books already on the shelf. The way we pick up the book is affected by many constraints *i.e.* the room in the box, the size of the gripper (*e.g.* our hand), and the book's size. On the other hand, the place operation poses further constraints *i.e.* the presence of other books, the location on the shelf, etc. Ideally, we would like to select the initial grasp such that we may perform the task without any issues. However, the combination of these constraints makes the selection of the grasp extremely complex.

The task of unloading a box of books shares an equal degree of complexities that many industrial operations. The stage following the demolishing of a nuclear site is storing the removed pieces in containers. Because of the hazardousness of the materials and the costly equipment to store these objects (*e.g.* barrels), careful considerations about the organisation within the barrel make a huge difference.

The sparing of equipment is only a part of the beneficial effect of intelligent storage. With information about the task, the robot may *e.g.* minimise re-manipulations of the object to fit the object into the barrel or avoid suboptimal space organisation. This impacts the time consumed for each operation and speeds up the usually overly long process of decommissioning.

These simple examples showcase a critical point: the grasp action is a purposing action; alongside the stability of a grasp, we must consider the task to perform afterwards beforehand.

However, researchers have predominantly studied the planning of handling actions as three separates problems:

- G0, *Reaching phase*: a manipulator approaches an object;
- G1, *Grasp phase*: the manipulator makes stable contacts on the object surface, called force-/form-closure grasping configuration;
- G2, *Post-grasp manipulation phase*: it moves the object and delivers desired object movements.

Fig. 6.1 depicts the task of composing the text *ERL* using objects with several shapes. The robot must grasp the green parallelepiped and pose it as a leg for the letter *L*. We highlight the three phases. The robot approaches the object from above (G0). We obtain two possible grasps candidates (G1). The robot carries out the task (G2). Because of the constraints to the following task (G2), the upper grasp candidate results not being suitable, and the robot collides with the table. On the other hand, the lower grasp candidate allows the robot some room for performing the placing.

In this chapter, we will study how to merge G1 and G2, and we will propose two techniques that extend the current state-of-art for grasping selection.

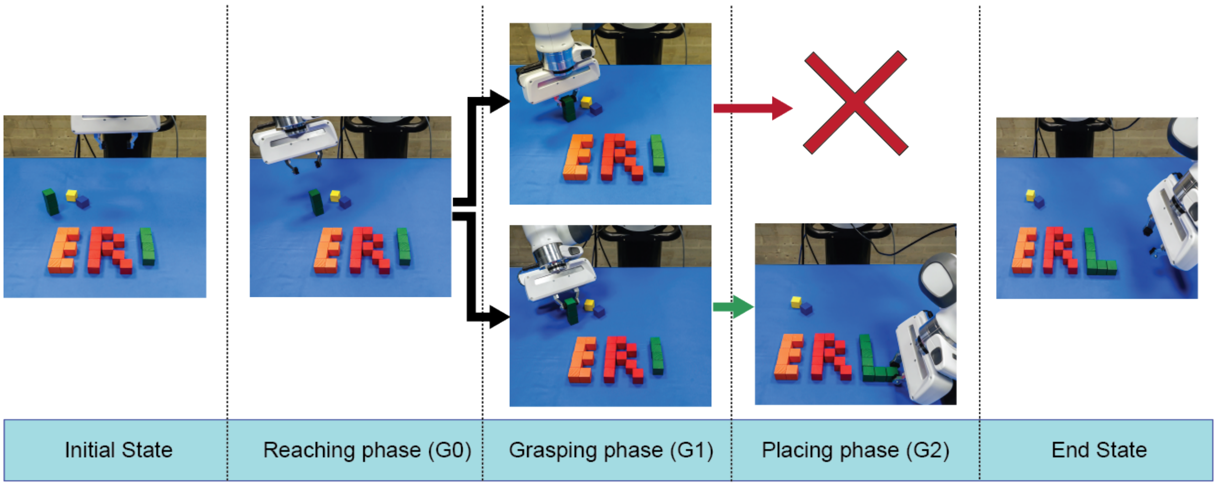


Figure 6.1: The robot is tasked with writing of the text ERL using blocks. In the foremost left picture, the robot is in initial state. At the reaching phase, the robot approaches the object and prepares to grasp. Two possible grasp candidates are available: from the top (top picture); from the side (bottom picture). The top grasp is not suitable because the robot would collide with the table. The side grasp allows the robot to pose the leg of the letter L in position. In the foremost right figure, the robot retracts.

6.2 Choosing grasp to enable collision-free post-grasp manipulation

None of the above work addresses the problem of selecting grasps that will permit a desired post-grasp manipulative trajectory of the target object, without any parts of the robot or object encountering collisions. However, many scenarios have obstacles that the robot must avoid to reach the position where the task takes place.

In this section, we address this problem by examining a number of possible stable grasping configurations on an object (G1). For each stable grasp, we explore the motion space of the manipulator which would be needed for post-grasp motions, to deliver the object along the desired trajectory, (G2). A criterion, based on potential fields in the post-grasp motion space, is used to assign a collision-cost to each grasp. A grasping configuration is then selected which enables the desired post-grasp object motion while minimising the proximity of all robot parts to obstacles during motion. We demonstrate our method with peg-in-hole and pick-and-place experiments in cluttered scenes, using a Franka Panda robot. Our approach is effective in selecting appropriate grasps, which enable both stable grasps and also desired post-grasp movements without collisions. We also show that, when grasps are selected based on grasp stability alone, without consideration for desired post-grasp manipulations, the corresponding post-grasp movements of the manipulator may result in collisions.

Contribution

The contribution of this section are:

- we compute a collision cost during G2 movements and use it to select a grasp candidate (G1) with the minimum collision cost for performing G2 movements;
- we show the effectiveness of our approach on three real manipulation tasks using a "Panda" robot manufactured by FRANKA EMIKA GmbH: push a peg in a barrel; screw a peg in the barrel; and pick-and-place.
- Our experimental results demonstrate that our proposed cost effectively differentiates grasping configurations resulting in a collision

6.2.1 Post-grasping collision avoidance metric

We assume a set of given grasp candidates is provided by a grasp generator algorithm. The robot we consider mounts a parallel jaw gripper. Many algorithms are present in the literature to generate possible grasp candidates onto the object surface *i.e.* [KDA+16b] or [tPGSP17].

Hereafter, we use [KDA+16b] to generate a set of grasp locations on the object:

$$^{r}\vec{x}_g = graspGenerator() \tag{6.1}$$

However, one can use any other state-of-the-art grasp synthesising approach.

We consider reference frame $\vec{x}_r \in SE(3)$ (the black frame in Fig. 6.2). Frame $^{r}\vec{x}_g \in SE(3)$ (in eq. 6.1) is attached to the end-effector at each time (shown with blue thick

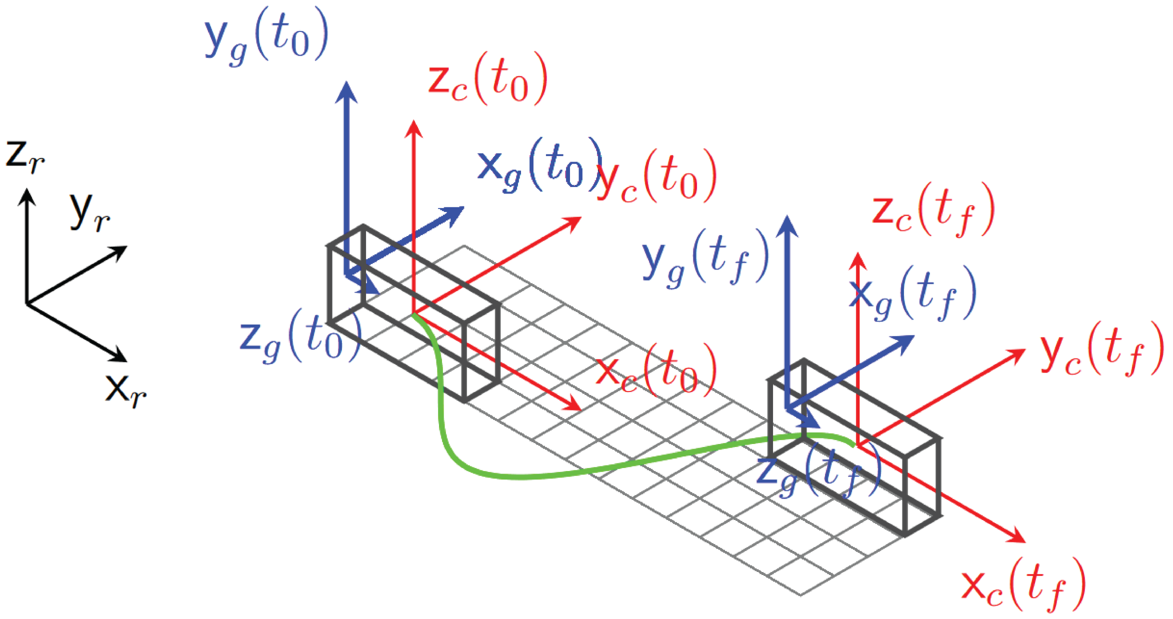


Figure 6.2: An object is shown in global coordinate frame $\vec{x}_r = \{\mathsf{o}_r, \mathsf{x}_r, \mathsf{y}_r, \mathsf{z}_r\}$ at initial and final time (t_0 and t_f) in the left and right side of the figure. Local coordinate frame ${}^r\vec{x}_c = \{\mathsf{o}_c, \mathsf{x}_c, \mathsf{y}_c, \mathsf{z}_c\}$ is attached to the centre of mass of the object, shown in red colour. This frame follows trajectory ${}^r\vec{x}_c(t)$ shown with green line during manipulative movements. The blue frame ${}^r\vec{x}_g = \{\mathsf{o}_g, \mathsf{x}_g, \mathsf{y}_g, \mathsf{z}_g\}$ represents a coordinate frame corresponding with a end-effector pose of one feasible grasp configuration at time t_0 and t_f

frames). We use operational space trajectory to refer to successive poses of this frame that correspond with a sequence of poses attached to the centre of mass of the object ${}^r\vec{x}_c = \{{}^r\mathsf{o}_c, \mathsf{x}_c, \mathsf{y}_c, \mathsf{z}_c\}$. ${}^r\vec{x}_c(t) = \{{}^r\mathbf{t}_c(t), {}^r\mathbf{R}_c(t)\; \forall\, 0 \le t \le \mathrm{T}\}$ defines a desired trajectory for the object, t denotes a time and t_f is time-to-completion for the manipulative movements, ${}^r\mathbf{R}_c(t)$ is rotation matrix from ${}^r\vec{x}_c(t)$ to $\vec{x}_r$, and ${}^r\mathbf{t}_c(t)$ represents translation of ${}^r\vec{x}_c(t)$ expressed in $\vec{x}_r$.

We assume the object is non-deformable. Hence, when the robot comes into contact with the object, a trajectory of the corresponding end effector pose at grasping configuration can be expressed based on ${}^r\vec{x}_c$ and a fixed transformation from ${}^r\vec{x}_g$ into ${}^r\vec{x}_c$, as follows:

$$\begin{aligned} {}^r\mathbf{R}_g(t) &= {}^r\mathbf{R}_c(t)\,{}^c\mathbf{R}_g \\ {}^r\mathbf{t}_g(t) &= {}^r\mathbf{t}_c(t) + {}^r\mathbf{R}_c(t)\,{}^c\mathbf{t}_g \end{aligned} . \tag{6.2}$$

where ${}^c\vec{x}_g = \{{}^c\mathbf{R}_g, {}^c\mathbf{t}_g\}$ are rotation and translation from ${}^r\vec{x}_g$ to ${}^r\vec{x}_c$, respectively. We consider the trajectory for the object movements, namely ${}^{r}\vec{x}_c(t)$, to be known, e.g. moving/opening a sliding door. Hence, for each grasping configuration, namely ${}^{c}\vec{x}_g$, the operational space trajectory of the end-effector, namely ${}^{r}\vec{x}_g(t) = \{{}^{r}\mathbf{t}_g(t), {}^{r}\mathbf{R}_g(t)\}$, is fixed and can be computed using eq.(6.2), i.e. ${}^{r}\vec{x}_g = {}^{r}\vec{x}_c {}^{c}\vec{x}_g$. An IK algorithm is then utilised to compute the corresponding joint space trajectory of the manipulator as follow:

$$\mathbf{q}(t|{}^{r}\vec{x}_c, {}^{c}\vec{x}_g) = \mathrm{IK}\left({}^{r}\vec{x}_g(t)\right) \tag{6.3}$$

where the vertical line in $f(\mathrm{y}|\mathrm{x})$ mean f is a function of y given x. In eq. (6.3), $\mathbf{q}(t|{}^{r}\vec{x}_g) = \{q_i(t) \quad | \quad q_i(t|{}^{r}\vec{x}_g) \in \mathbb{R}, i = 1, ..., n_q\}$ at each time step and n_q is the number of joints

We consider several body points attached to the manipulator's links ($b_j = \{b_{j,1} \in \mathbb{N}, b_{j,2} \in \mathbb{R}\}$ where $b_{j,1}$ denotes the link number to which the body point is attached and $b_{j,2}$ represents the corresponding distance between the j_{th} joint and the body point.), e.g. one body point is attached to every joint and one is attached to the middle of every link. Thus, the trajectory of each body point is:

$$\mathbf{b}_j(t|{}^{r}\vec{x}_g) = \mathrm{FK}_j\left(\mathbf{q}(t|{}^{r}\vec{x}_g)\right), \; j = 1, ..., n_{bp} \tag{6.4}$$

where n_{bp} is the number of body points and FK_s is the forward kinematics showing the position of j_{th} body point, namely $\mathbf{b}_j \in \mathbb{R}^3$, at each time step. A group of trajectories expressing the movements of the body points, as per eq. (6.4), for a known grasping candidate ${}^{r}\vec{x}_g$ and a given sampling time of a desired post-grasp trajectory is

$$\begin{aligned} \mathrm{X}|_{{}^{r}\vec{x}_g} = \{\mathbf{b}_j(k) \quad | \quad \mathbf{b}_j(k)|_{{}^{r}\vec{x}_g} = \mathrm{FK}_j\left(\mathbf{q}(k)|_{{}^{r}\vec{x}_g}\right); \\ j = 1, ..., n_{bp}; k = 1, ..., K\} \end{aligned} \tag{6.5}$$

where $f(\mathrm{y})|_{\mathrm{x}}$ means f is a function of y given x, $\mathrm{X}|_{{}^{r}\vec{x}_g} \in \mathbb{R}^{n_{bp} \times K}$, n_{bp} is the number of body points and K represent the total number of discrete time steps k. For the sake of simplicity, we write $\mathbf{b}_j$ instead of $\mathbf{b}_j(k)|_{{}^{r}\vec{x}_g}$ in the following.

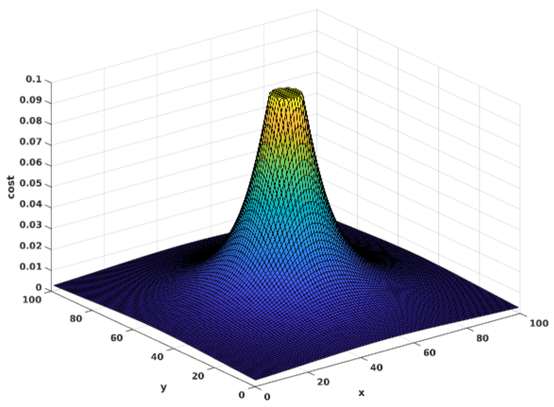


Figure 6.3: Example of the Hwang's potential field for the constraint $g(x, y) := x^2 + y^2 = 1$ and $\delta = 10$. The value within the constraint, $x^2 + y^2 < 1$, are δ^{-1}, whereas the outside cost decrease sharply with the distance to the edge of the circle.

We specify an obstacle by a set of linear constraints as follows:

$$g_h^i(\mathrm{x}) \leq 0, g_h^i \in L^m, \mathrm{x} \in \mathbb{R}^3 \tag{6.6}$$

This expresses a set of inequalities describing a convex region including i_{th} obstacle where Ls are linear functions.

[HA92] proposed a function that is arbitrarily large in the region inside the obstacle and decreases sharply proportional to d that is the distance between $\mathbf{b}$ and $g_h(\mathrm{x}) = 0$ using eq. (6.5) and (6.6), Fig. 6.3.

$$d_i\left(\mathbf{b}_j, \mathrm{x}\right) = \sum_{h=1}^{n_h} \{\hat{d}\left(g_h^i(\mathrm{x}) = 0, \mathbf{b}_j\right) + \left|\hat{d}\left(g_h^i(\mathrm{x}) = 0, \mathbf{b}_j\right)\right|\} \tag{6.7}$$

where n_h is the number of linear constraints and $\hat{d}$ is the distance between $\mathbf{b}$ and the plane $g_h(\mathrm{x}) = 0$. If $\mathbf{b}_j$ is on or inside the convex region of the i_{th} obstacle, then $d_i \leq 0$. Hence, we express the costs for all body points, obstacles and sample points of G2 by using eq. (6.6) and (6.7) as follows:

$$C_i({}^r\vec{x}_g|g_h) = \begin{bmatrix} c^i_{11} & c^i_{12} & c^i_{13} & \cdots & c^i_{1T} \\ c^i_{21} & c^i_{22} & c^i_{23} & \cdots & c^i_{2T} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c^i_{n_{bp}1} & c^i_{n_{bp}2} & c^i_{n_{bp}3} & \cdots & c^i_{n_{bp}T} \end{bmatrix} \tag{6.8}$$

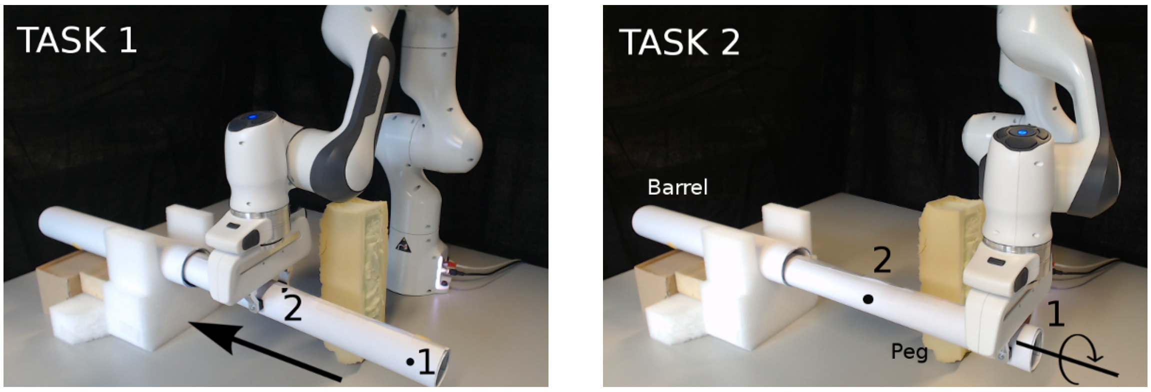
Where $c^i_{j,k}|_{{}^r\vec{x}_g,g_h}$ represents the cost $d_i(\mathbf{b}|{}^r\vec{x}_g, g_h)$ for the i_{th} obstacle, j_{th} body point at k_{th} time step. We can easily extend our cost computation to more than one obstacle $C = [C_1, C_2, ..., C_{n_o}]$ where n_o is the number of obstacles. Finally, eq. (6.9) defines the collision cost for a desired post-grasp trajectory of the object and an obstacle. Given the desired trajectory of the object, as per eq. (6.2), and an obstacle as per eq. (6.6), we can write $C({}^r\vec{x}_g, g_h)$ in eq. (6.8) as a function of ${}^c\vec{x}_g$, i.e. $C({}^c\vec{x}_g|{}^r\vec{x}_c, g_h)$. Hence, we write the obstacle avoidance cost $J(C({}^r\vec{x}_g, g_h))$ as a function of ${}^c\vec{x}_g$, as per eq. (6.9).

$$J({}^c\vec{x}_g|{}^r\vec{x}_c, g_h) = |\underline{\mathbf{1}} \oslash (C({}^c\vec{x}_g|{}^r\vec{x}_c, g_h) + \underline{\mathbf{1}}\delta)|_1 \tag{6.9}$$

where δ^{-1} is a fixed value for setting the maximum available cost, $\underline{\mathbf{1}}$ is the all-ones matrix, and $\oslash$ Hadamard division operator (*i.e.* element wise division operator). J yields maximum costs on and inside the convex region representing an obstacle. A post-grasp manipulator trajectory that yields minimum cumulative value of cost ideally allows collision-free movement for the robot. By definition, the value of J is a function of grasping configuration, obstacle position and shape and object trajectory.

$${}^c\vec{x}^*_g = \underset{{}^c\vec{x}_g}{\operatorname{argmin}}\ J \tag{6.10}$$

Therefore, we select the optimal grasp candidate ${}^c\vec{x}^*_g$ that minimises collision cost for the given post-grasp movements and obstacle position.



(a) (b)

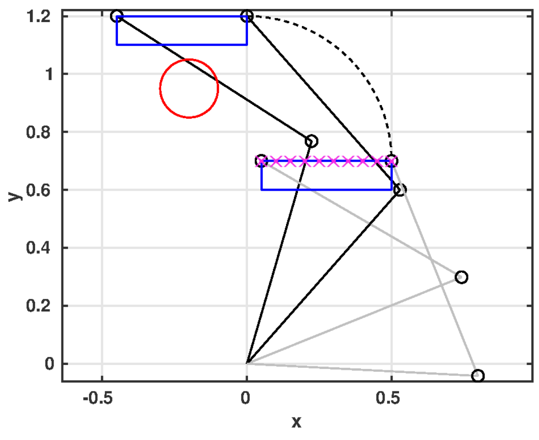
Figure 6.4: The Panda is tasked with inserting and rotating the cylindrical peg into the cylindrical shell. The manipulator grasps the peg at 1 ((b)) which is suitable for rotating clockwise the peg $\frac{\pi}{6}$ [rad]. However, it is not appropriate for inserting peg 0.15 [m] into the barrel. By contrast, while the manipulator can insert the peg 0.15 [m] into the shell by grasping it at 2, it is not able to rotate the peg clockwise because of the collision.

6.2.2 Experimental validation

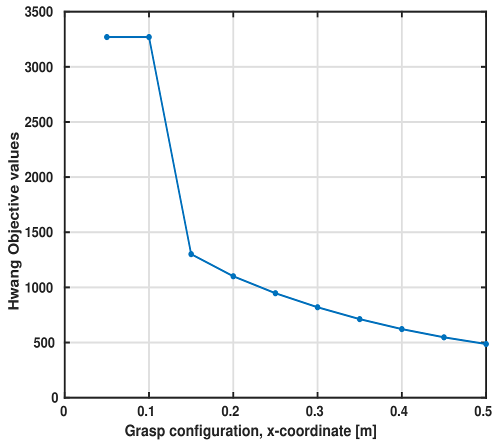
First, we validate our approach on a simple 2D scenario using a robot with two revolute joints, which must move a rectangular object. We consider *a priori* knowledge of a number of grasp candidates on the top edge of the rectangle and the trajectory of the object, Fig. 6.5(a). Furthermore, we set up a "Panda" manipulator manufactured by FRANKA EMIKA to perform experiments showing the effectiveness of our approach for selecting a grasping configuration. The Panda has 7 revolute joints and an electric parallel jaw gripper. We designed three experiments:

1. pushing a peg into a barrel, Fig. 6.4(a);
2. rotating the peg in the barrel, Fig. 6.4(b);
3. pick-and-place task (Fig. 6.7)

First, the Panda is tasked with pushing and rotating the cylindrical peg in the barrel.This is a common task in the industry in which the trajectory of the object is constrained and known prior to G0 and G1. Two stable grasps are considered, Fig. 6.4, that have high likelihood as per eq. (6.1). We use Trac-IK [BA15] and eq. (6.2) to compute



(a)



(b)

Figure 6.5: (a) 2-D planar manipulator moves the cuboid object from the bottom right to the top left along the black dashed line. The red circle represents an obstacle added to the workspace. (b) shows the cost corresponding with each grasping candidate at the initial position of the object. The grasps at 0.05 [m] and 0.1 [m] yield collision, hence, they result in the maximum cost, *i.e.* δ^{-1}.

the trajectory of the end-effector corresponding with G2 (shown with the black arrows in Fig. 6.4(a) and 6.4(b)) and grasp 1 and 2.

We also analysed the time to compute the cost function, and broken-down the elapsed time to identify possible bottlenecks of our approach.

2-D link simulated manipulator

Fig. 6.5(a) shows a 2-D manipulator tasked with grasping a rectangle object from its top edge and moving it from the bottom right to the top left of the image. Nonetheless, the grasps at the far left part of the top edge of the object result in collisions between the link of the manipulator and an added obstacle, shown by the red circle. As such, from the set of feasible grasping configurations (shown by green crosses on the blue rectangle in Fig. 6.5(a)), just a few numbers of them allow the manipulator to perform the desired post-grasp movements. Therefore, the robot must select the one that yields overall collision-free movements.

For each grasping configuration, an obstacle avoidance cost is estimated using eq. (6.9).

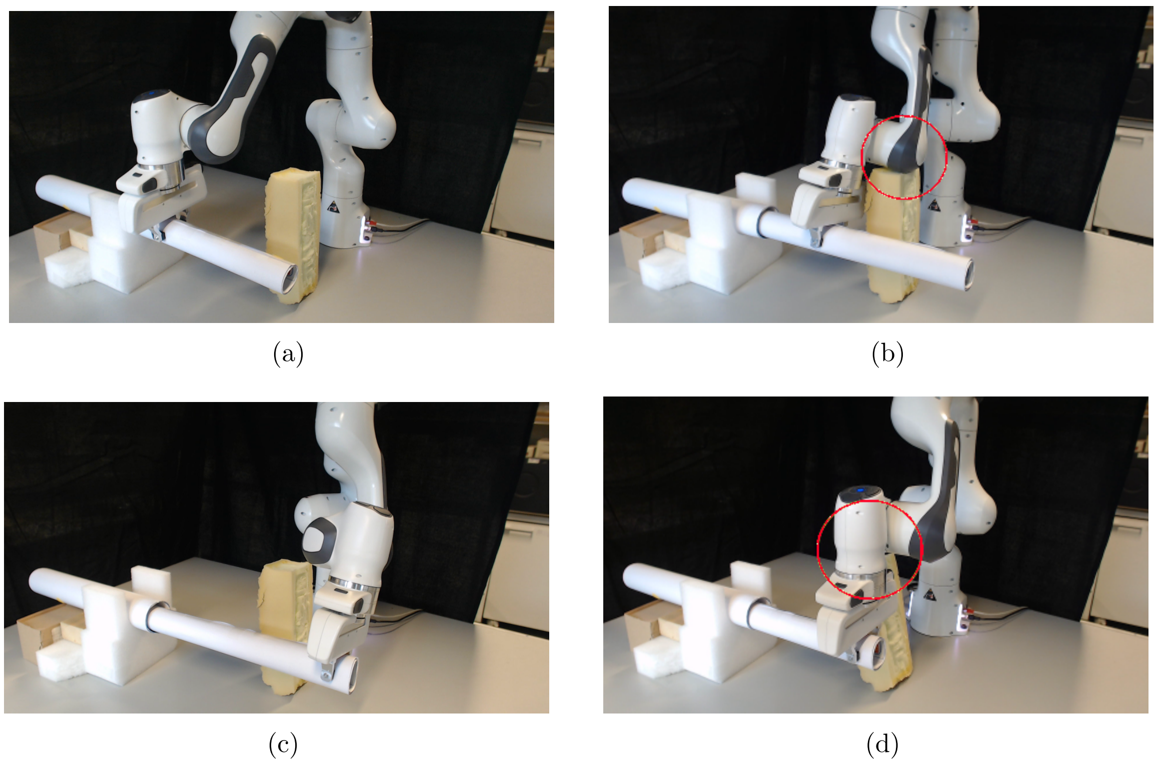


Figure 6.6: This figure shows the Panda performs pushing ((a) and (b)) and rotating ((c) and (d)) by grasping the cylindrical peg at 1 and 2 (shown in Figs. 6.4(a) and 6.4(b)). Figs. (a) and (b) show the Panda performed the inserting and rotating task without colliding with the obstacle while it grasped the peg at 1 and 2, respectively. On the contrary, grasp 1 and 2 resulted in a collision during performing pushing and rotating task, respectively (Figs. (b) and (d)).

Fig. 6.5(b) shows the estimated costs for selecting a grasp configuration and performing the given task. The lower the cost, the larger the distance between the manipulator and the obstacles during movements. In 6.5(b) the vertical axis shows the obstacle avoidance cost J and the horizontal axis is the x coordinate of the grasping configuration on the top edge of the blue object. Each grasping configuration is at 0.05 [m] from another. The cost values suggest that grasping configurations at the far right of the object (i.e. $\mathrm{x} = 0.5[m]$) is the most suitable one for moving the object along the given trajectory.

Real robot experiments

In the first task, the manipulator pushes the peg into a cylindrical barrel by moving the peg from right to left, as shown in Fig. 6.6(a). In the second task, the manipulator rotates the peg clockwise in the barrel by $\frac{\pi}{6}$ [rad] (Fig. 6.6(c)). An obstacle in the workspace, however, makes one of the grasp unsuccessful as it results in collision between the manipulator and the obstacle. The collision costs are $J_{(t_1,g_1)} = 11126$, $J_{(t_1,g_2)} = 3512$, $J_{(t_2,g_1)} = 6887$ and $J_{(t_2,g_2)} = 15318$ where t_* and g_* represent $*_{th}$ task and grasp, respectively.

These cost values indicate that grasp 2 results in collision-free movements during performing task 1, as shown in Fig. 6.6(a), while it collides with the obstacle during executing task 2 as shown in Fig. 6.6(b). On the other hand, grasp 1 results in successful completion of task 2 (Fig. 6.6(c)) while yields collision during post-grasp movements of task 1 (Fig. 6.6(d)). This shows that the cost values correctly predict collisions during post-grasp movements.

Next, the Panda is tasked with picking up a white cuboid object (located on the table in front of the robot) and placing it at the desired target pose shown with G in Fig. 6.7(a)). The desired trajectory corresponding with the object movements (G2) is shown with a red dashed line and is known prior to the experiment. A set of 6 grasping configurations with high likelihood are also considered (using eq. (6.1)).

We add an obstacle to the robot's workspace. We capture the obstacle position by an Astra Pro RGB-D sensor, which tracks an Aruco marker, and we convert the obtained position in the robot frame using the camera calibration transformation. As the collision-free object movements are known, only a collision between the manipulator and the obstacle may cause a problem during movements. We compute the collision cost for all 6 grasps (shown in Fig. 6.7(a)) which are shown in Fig. 6.7(b). These costs show that grasps 5 and 6 are collision-free (Fig. 6.7(d)) and 1 to 4 yield collision between the manipulator and yellow cuboid during post-grasp movements (Fig. 6.7(c)). We performed the desired movements and Fig. 6.7(c) and 6.7(d) show the manipulator during post-grasp movements with unsuccessful (grasp 1) and successful (grasp 6) grasping configurations, respectively.

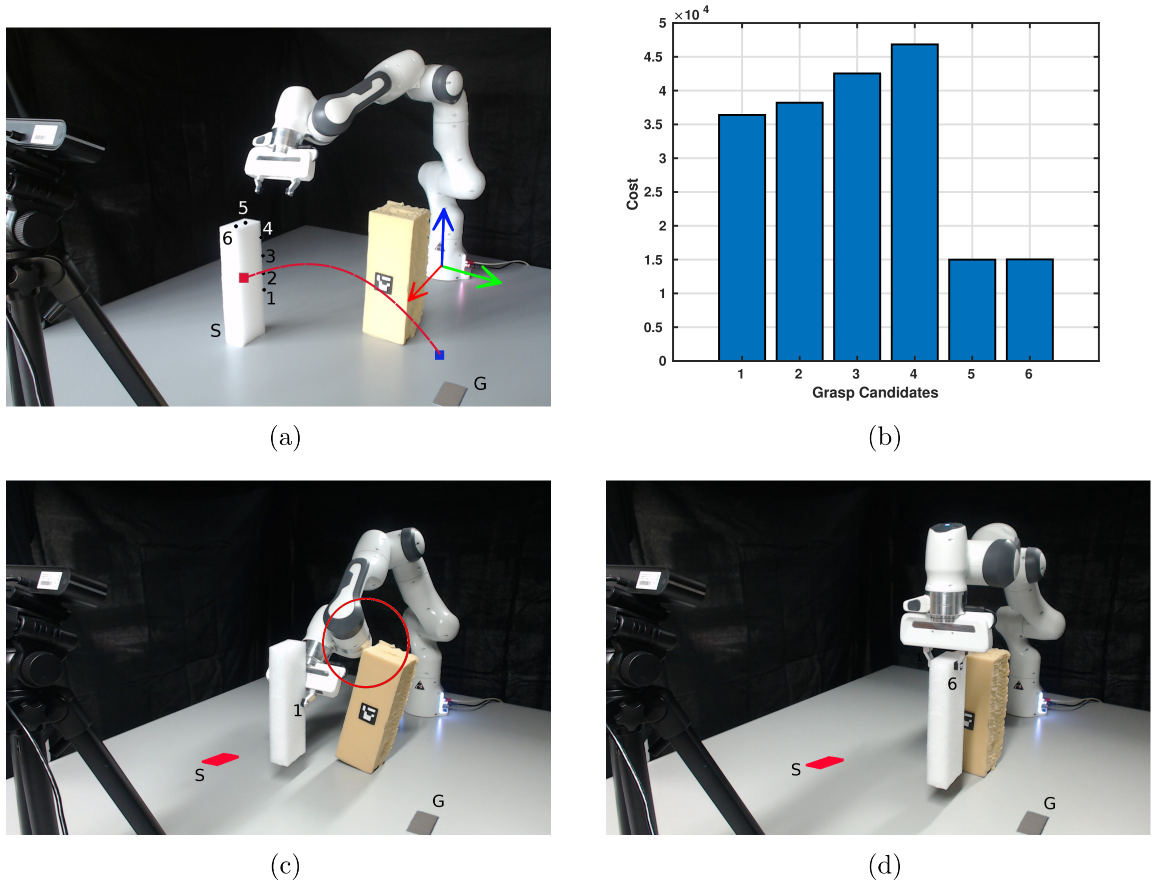


Figure 6.7: Experimental set up: (a) the Panda robot picks up the white cuboid object at initial position S and places it on grey rectangle G. The movement trajectory is assumed to be given and is shown with the red dashed line. The robot reference frame is shown with RGB colour corresponding with $\{\text{X}, \text{Y}, \text{Z}\}$. A marker-based tracking system computes the position of the obstacle in the robot reference frame. Based on this position a cost for each grasp candidates is evaluated as per eq. (6.9). (c) the robot collides with the obstacle (marked with the red circle) as it randomly selected the grasping configuration without using our predictive grasping approach; (d) the robot successfully performs the task by using our predictive grasp planning approach. (b)collision costs corresponding with each grasp candidate of the pick-and-place task shown in Fig. 6.7(a).

6.2.3 Time evaluation

The time for computing 10K IK and the obstacle avoidance cost is shown in Fig. 6.8. The mean value of IK and the cost computation time is 0.0104 [s] and 0.0107 [s], respectively. Although the maximum time of computing IK and J are 0.0177 [s] and 0.0171 [s], these represent outliers of data. 25_{th} and 75_{th} percentiles of the computation time samples are in the interval of 0.0102 [s] and 0.0103 [s] for IK and 0.0105 [s] and 0.0107 [s] for the

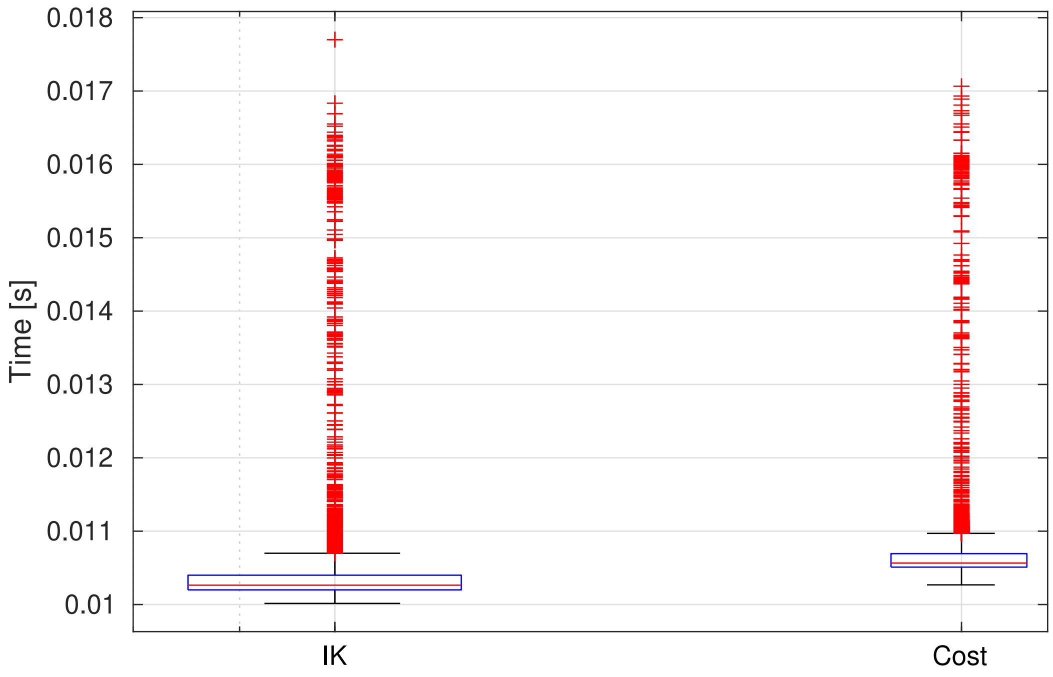
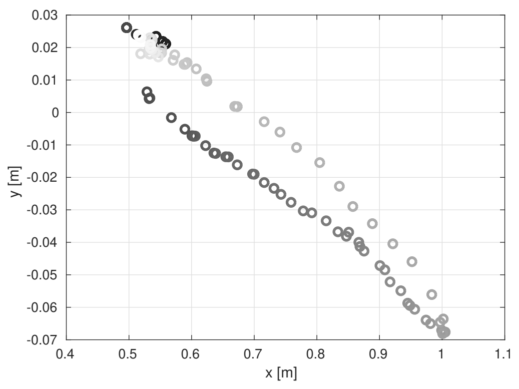
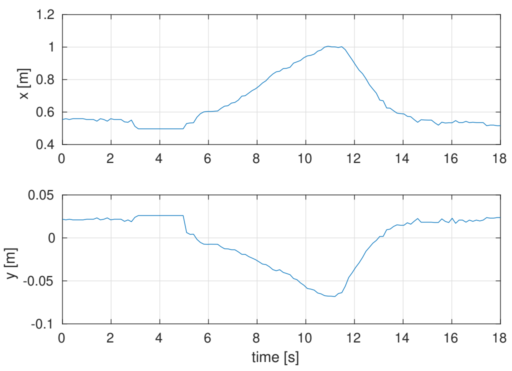


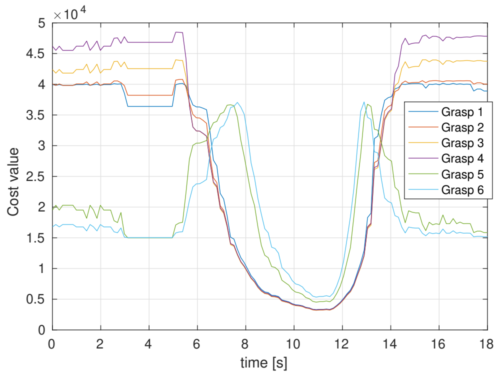
Figure 6.8: Time of computing 10,000 times the IK and the obstacle avoidance cost values. The mean value of computation time is 0.0104 and 0.0107 [s] for IK and cost, respectively. The post-grasp trajectory includes 100 sample points. Nonetheless, the number of sample points of a post-grasp trajectory can be significantly reduced as long as the distance between two consecutive sample points is less than the minimum dimension of the obstacle, e.g. length, height and width of a cuboid. This allows us to be sure there is at least one sample capturing a collision.

whole cost computation. This includes computing the IK and obstacle avoidance cost. To sum, our expected mean and median cost computation frequency are 93 [Hz] and 94 [Hz], respectively. In the worst case, the expected cost computation frequency is 58 [Hz].

A further experiment shows the correlation between the obstacle position and the computed cost. Here, a user moves an obstacle in the robot's workspace while our proposed approach computes the collision cost. Fig. 6.9(a) and 6.9(b) show the trajectory of the obstacle moved by a human. Fig. 6.9(c) depicts the corresponding obstacle avoidance costs for all six grasp configurations at each time step. Three segments are identifiable throughout the trajectory based on costs in Fig. 6.9. In the first and the last segments of the trajectory, grasping configurations 3, 4, 5 and 6 yield high costs that indicate possible


(a)


(b)



(c)

Figure 6.9: These figures show the computed cost values for performing the pick-and-place task with the Panda robot. (a) and (b) shows the trajectory of the obstacle that is moved by a human during the experiment. (c) shows evaluated values for each grasp configurations during the experiment. These figures demonstrate how the values of costs at different grasping configurations vary with the change in position of the obstacle. In the middle part of the experiment, the obstacle is outside the robot's workspace, which yields a drop in all the cost values. This makes even grasps 5 and 6 less costly than grasps 1 to 4. However, if the obstacle is in the way of manipulative movements, as per $t = 0 - 5[s]$ to $t = 14 - 18[s]$, grasps 5 and 6 yield a very large value of costs.

collision of robot and obstacle, whereas they obtain a minute cost values in the middle part of the trajectory. On the other hand, grasps 1 and 2 yield collision-free post-grasp movements during the initial and final parts of the obstacle trajectory.

6.2.4 Conclusion

This section presented a method for selecting a grasp enabling post-grasp manipulative trajectories to be achieved without collisions. This work complements other work on algorithms for grasp synthesis. In principle, any grasp synthesis method can be combined with this work in a modular way, *i.e.* our method will select between a variety of stable grasps proposed by an arbitrary grasp synthesis method and choose the best to enable the post-grasp manipulation. Our method computes the collision cost corresponding to each proposed grasp that would incur by the manipulator to deliver the target object along its desired post-grasp trajectory.

Previous studies have predominantly considered collision-free reach-to-grasp trajectory planning (G0) and forming a stable grasp on an object (G1). In contrast, this paper has studied how intelligent grasp selection enables collision-free post-grasp movements (G2). We have demonstrated the effectiveness of our approach in three different common robotics tasks: (1) pushing a cylindrical peg inside a tube, (2) rotating the peg inside the tube, and (3) a pick-and-place task. Our results show how a robot can intelligently select a grasping configuration by analysing possible collisions during post-grasp motions.

A statistical analysis of computation time suggests that we can use this system to guide a human operator (using a haptic teleoperation device) by providing real-time haptic cues towards a grasp that is collision-free during known post-grasp trajectories. For this application, a cost-computation frequency greater than 15 [Hz] was shown to be sufficient for providing the real-time perception of haptic cues to humans. In this paper, our empirical experiments showed that the frequency of computing collision costs on a standard 5-core Intel PC is greater than 50 [Hz] in the worst case. Since our method is based on computations readily parallelisable, the approach is computationally scalable to address the case of more obstacles and highly cluttered scenes in real-time. This is particularly important in constrained environments, where a sample-based approach randomly explores a high-dimensional space, which might require more than a few seconds to find a solution. On the other hand, our approach yields a solution in a few milliseconds.

6.3 Multi-criterion selection for post-grasp manipulation

As we have aforementioned, the grasp selection process is gaining momentum, Sec. 6.2.1, and other works have investigated this point of view on the same problem.

Recently a bulk of papers, [MAS$^+$16, MGER17, GMS17], addressed the *post-grasp* phase (G2) and shared the idea of selecting a grasp candidate in order to minimise some specific objective demanded by the task. In [MGER17], a method picks the grasp configuration that is more suitable to minimise the impact on external obstacles. In [MAS$^+$16] and [GMS17], the authors propose a grasp candidate selector, which enables torque-efficient manipulations and facilitates G2 motions using a Task-Relevant Velocity Manipulability (TOV), respectively.

Although these metrics investigate several aspects of the G1/G2 phase, they focus on a single component. However, many applications demand more than one of these objectives be minimised at the same time. In this section, we considered four different grasp selection costs and investigated the impact of accounting for multiple objectives in the G1/G2 phase.

Contribution

- we consider four grasp selection costs, and we formulate a minimisation problem;
- we evaluate on a simple scenario, the impact of the weights on the minimisation problem;

6.3.1 Task-informed post-grasping approaches

We consider reference frame $\vec{x}_r \in SE(3)$ (the black frame in Fig. 6.2). Frame $\vec{x}_g \in SE(3)$ is attached to the end-effector at each time (shown with blue thick frames). We use operational space trajectory to refer to successive poses of this frame that correspond with a sequence of poses attached to the centre of mass of the object $\vec{x}_c = \{{}^r\mathsf{o}_c, \mathsf{x}_c, \mathsf{y}_c, \mathsf{z}_c\}$. $\vec{x}_c(t) = \{{}^r\mathbf{t}_c(t), {}^r\mathbf{R}_c(t)\ \forall\, 0 \le t \le \mathrm{T}\}$ defines a desired trajectory for the object, t denotes a time and t_f is time-to-completion for the manipulative movements, ${}^r\mathbf{R}_c(t)$ is rotation matrix from $\vec{x}_c(t)$ to $\vec{x}_r$, and ${}^r\mathbf{t}_c(t)$ represents translation of $\vec{x}_c(t)$ expressed in $\vec{x}_r$.

We assume the object is non-deformable. Hence, when the robot comes into contact with the object, a trajectory of the corresponding end effector pose at grasping configuration can be expressed based on $\vec{x}_c$ and a fixed transformation from $\vec{x}_g$ into $\vec{x}_c$, as per 6.2. We consider the trajectory for object movements, namely ${}^r\vec{x}_c(t)$, is known, *e.g.* moving/opening a sliding door. Hence, for each grasping configuration, namely ${}^c\vec{x}_g$, the operational space trajectory of the end-effector, namely $\vec{x}_g(t) = \{{}^r\mathbf{t}_g(t), {}^r\mathbf{R}_g(t)\}$, is fixed and can be computed using eq. (6.2). An IK algorithm (TRAC_IK, [BA15]) is then utilised to compute the corresponding joint space trajectory of the manipulator as follow:

$$\mathbf{q}(t)|_{{}^c x_g, x_c} = \mathrm{IK}\left(\vec{x}_g(t)\right),$$

where $\mathbf{q}(t) = \{q_1(t), ..., q_{n_q}(t)\}$, $q \in \mathbb{R}$ at each time stamp and n_q is the number of joints.

6.3.2 Collision avoidance objective - J_c

We exploit the previously defined collision avoidance cost for grasping selection, Sec. 6.2.1.

We report the cost for sake of readability:

$$J_C = |\underline{\mathbf{1}} \oslash (C + \underline{\mathbf{1}}\delta)|_1 \tag{6.11}$$

where C is the matrix containing costs based on the distance between body points and obstacles, δ^{-1} is a fixed value for setting the maximum available cost, $\underline{\mathbf{1}}$ is the all-ones matrix, and $\oslash$ is the Hadamard division operator. J_c yields maximum costs on and inside the convex region representing an obstacle.

6.3.3 Torque objective - J_T

We assume that the dynamic model of the robot is known, and we have the whole motion equations of the manipulator. To find a grasp at minimum effort, the "augmented" equation of motion is used. Thanks to this formulation, we obtain a set of motion equations describing the dynamics of the robot with the object attached to its end-effector, [MAS$^+$16]. The effort metrics are computed in terms of the L_2 norm of the joint torques, as follows:

$$J_T = \frac{1}{T}\sum_{i=1}^{i=T}\left(\left|\frac{\tau(t_i)+\tau(t_{i+1})}{2}\Delta t\right|^2\right) \tag{6.12}$$

6.3.4 Safe Objective - J_S

Assuming, as in 6.3.3, to know the model of both robot, and the object we want to manipulate. Using the "augmented" equation of motion, the inertia matrix of the robot plus object is retrieved.

$$\Delta_{TOT} = \Delta_{Robot} + \Delta_{obj} \tag{6.13}$$

To obtain a safe grasp, the robot has to select the grasp configuration that yields minimum kinetic energy. [MGER17] proposes to minimise the total mass along all the trajectory.

$$J_S = \frac{1}{v^T \Delta_{TOT}^{-1}(x) v} \tag{6.14}$$

where v is the unity vector in the direction of motion.

6.3.5 Manipulability objective - J_M

A well-known indicator for the manipulation property of a non-redundant robot is the manipulability ellipsoid, described by the equation:

$$u^T (JJ^T)^{-1} u = 1 \tag{6.15}$$

where u is the end-effector twist, and J the robot's Jacobian. In order to maximise the manipulability ellipse along all robot post-grasp movement, we can minimise such quantity, [GMS17]:

$$J_M = \int_{t_0}^{T} \frac{1}{a^2(t)} dt \tag{6.16}$$

where $a(t)$ is the average ellipsoid radius at time t.

6.4 Objectives Optimisation

By construction, each objective depends on the grasp configuration and yields best results with lower values. We therefore define a matrix $\mathrm{J} \in \mathbb{R}^{4 \times n_{gc}}$ with 4 rows and n_{gc} columns, number of grasp configurations, as follows:

$$\mathrm{J} = [J_C \ J_T \ J_S \ J_M]^T \tag{6.17}$$

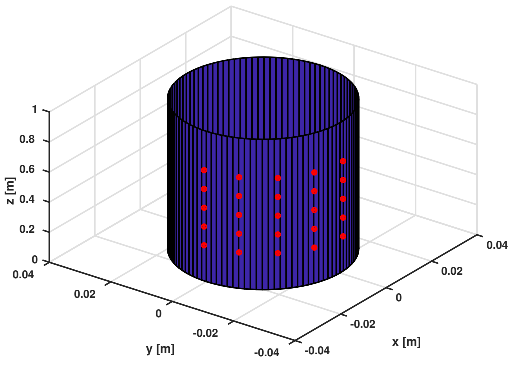
Then, we want to select the best ${}^c\vec{x}_g$ over the set of available grasp configurations:

$${}^c x_g^* = \underset{{}^g x_c}{\operatorname{argmin}} \ W \ \mathrm{J} \tag{6.18}$$

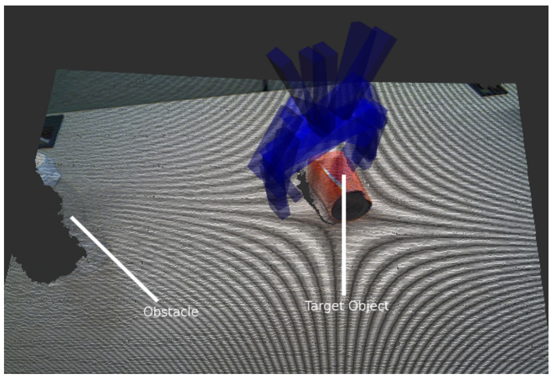
where $W \in \mathbb{R}^4$ is a matrix of coefficients for scaling objectives priority depending on the task, and normalises all magnitudes.

6.4.1 Experiments

The proposed approach is tested on two experimental setups. A simulated model of a Panda Robot, manufactured by Franka Emika, has been used in both experiments. It has 7 Degree of Freedom (DOF), and its kinematic model is provided directly by the manufacturer. In the first experiment, the robot is tasked with a pick and place operation of a cylindrical object, and all grasping configurations are hard-coded upon the surface of



(a)



(b)

Figure 6.10: 6.10(a) shows the grasp configurations on the object's surface for the first experiment, 6.10(b) depicts the setup for the second experiment; the robot picks the orange mug by one of the proposed grasping configuration (blue poses are sample generated by GPD), then it follows a trajectory which is a straight line toward the obstacle on the left.

the object; the task in the second experiment is again a pick and place operation, but the grasping configurations are provided by a neural network based algorithm, GPD, which selects a set of best grasp configurations in the scene.

As explained in section 6.4, a scaling vector is needed in order to obtain good performance for the optimisation problem. In this paper, we decided to use the standard score for scaling different objectives' magnitude.

$$W = [w_1 \ w_2 \ w_3 \ w_4] \tag{6.19}$$

$$w_i = \frac{J_i - \mu(J_i)}{\sigma(J_i)} \tag{6.20}$$

As in 6.20, the standard score maps mean values to zero and scales costs by the objectives' variance. An important property of this technique is that all negative values infer better results compared to the mean, and positive amounts mark worst results than the average. We remind that all objectives are designed to obtain values inversely proportional to better results.

6.4.2 Simulation experiment

The robot has to grasp a cylindrical object and move it along a predefined post-grasp trajectory. A collection of 25 grasp configurations upon the surface of the object is given, Fig.6.10(a). They describe the end-effector's pose (cartesian position and quaternion) to reach for performing a good grasp. The goal of the algorithm is to select the one grasping configuration for maximising all objectives together. In Fig.6.11(a), 6.11(b),6.11(c), 6.11(d) are showed the objectives' cost for each single grasping configuration; collision avoidance, manipulability, torque and safety respectively. The red colour in the graphs represents the best grasping configuration when a single objective is taken into consideration. However, as a result of a lower value in the global optimisation process, the selected grasping configuration is the $21th$, Fig.6.11(e).

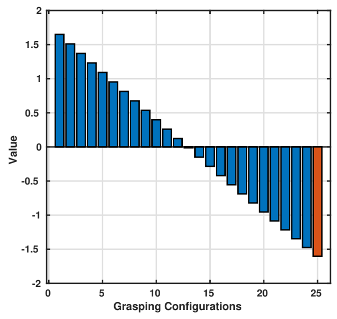
6.4.3 Real scenario experiment

The second experiment wants to show how this technique can be successfully extended to different grasping configurations generator. In the first stage, the neural network based GPD algorithm generates a collection of grasping candidates upon the surface of an orange mug made of plastic. Then, these configurations are passed to our optimisation algorithm, which selects the best candidate according to the objectives' values.

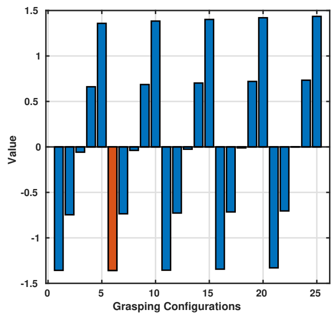
Fig.6.10(b) shows a few of the 43 grasping configurations generated by GPD, and, like for the previous experiment, costs for all objectives are reported in Fig.6.12. Moreover, in Fig.6.12(e) the resulting best grasp selected by the optimisation process is shown in red.

6.4.4 Conclusion

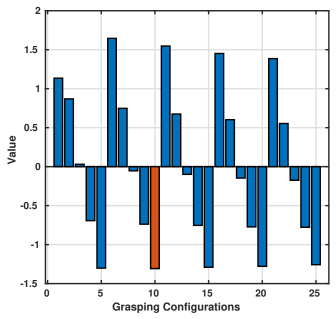
This section presents a novel predictive grasping approach that mixes four objective functions. The approach selects a grasping configuration from a set of feasible ones. This grasp yields the best values for collision-free, subsequent manipulative movements, manipulability, safety, and torque-efficiency.



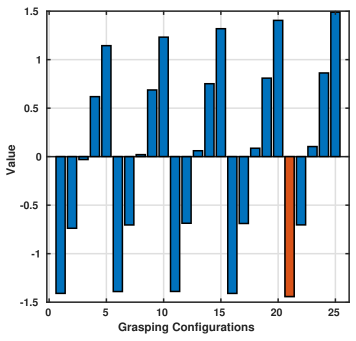
(a)



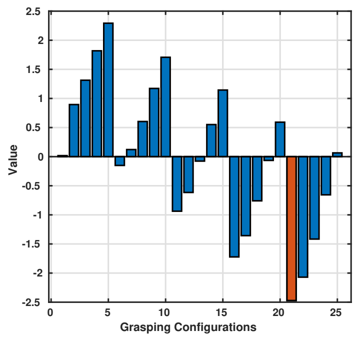
(b)



(c)



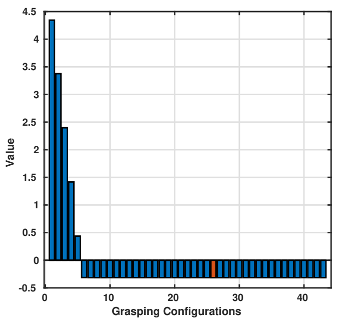
(d)



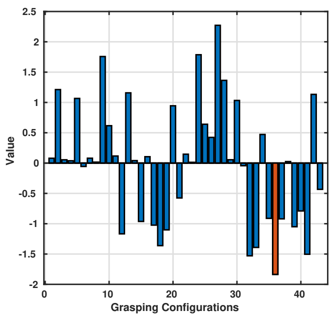
(e)

Figure 6.11: In these pictures, the object's cost for moving the object from an initial pose to a final goal is depicted for each objective. The normalisation technique move the average at zero and assigns positive numbers for worst results, and negative numbers to smaller costs. The red bar in all graphs shows the winning grasping configuration when a single objective is taken into consideration Collision Avoidance, Manipulability, Torque, and Safety consecutively. Fig.6.11(e) shows the result of the optimisation, and the red bar is the global winning grasping configuration.

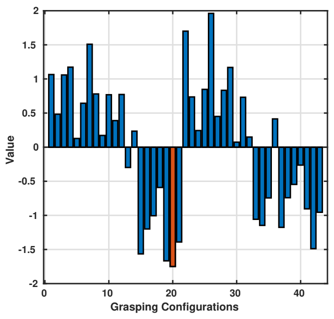
Grasping an object and manipulating it were studied typically in isolation in the previous works. For instance, [AtPP17] studied how to form stable contacts between a robot's hand and object surface. Although this approach is proper for laboratory purposes,



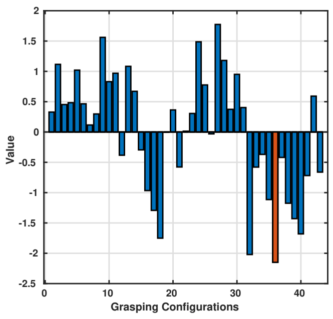
(a)



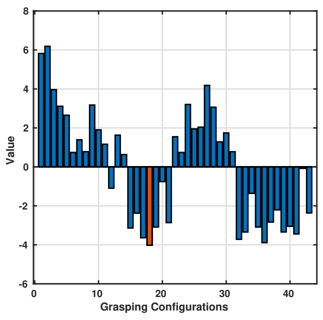
(b)



(c)



(d)



(e)

Figure 6.12: The costs for moving the target object from its initial position to the goal are depicted in the first four figures. The object follows a predefined collision-free trajectory toward the obstacle without yielding any collision with it. However, depending on the grasping configuration, the robot can collide with an obstacle performing the predefined post motion. Left grasping configurations in Fig.6.12(a) show this behaviour, and they yield high-cost value. Of the 43 grasp candidates, the best solution is the 18*th*.

it cannot be applied to real-world scenarios or can lead to underperformant results. For example, the robot may collide with obstacles during manipulative movements because of the chosen grasp or perform a high-torque trajectory to perform the task. Our novel grasp selection approach foresees such situations and selects the best grasp among a set of given grasp configurations which yields a collision-free movement, a torque-efficient trajectory, low impact on the outside object, and maximises manipulability ellipse.

We propose a methodology for merging all objectives together, and we provide some coefficients for tuning the optimisation process and weighting a specific objective.

Here, we assume the trajectory for collision-free movements of an object is planned regardless of the manipulator kinematics. In many real-world problems, this assumption is very useful, and it yields reduced complexities. For instance, a book trajectory can be provided by a learn from demonstration approach or assigned because of structural constraints (*e.g.* a sliding door). Hence, we built our approach based on the given object trajectory. Here, we considered the case where the set of grasp configurations is composed of a discrete number of candidates. In future works, we will relax this assumption and make a continuous manifold around the grasp configurations. Furtheremore, we will also consider more sophisticated optimisation processes to improve the results.

CHAPTER 7

Dual-arm task-informed manipulation

7.1 Introduction

This chapter explores the class of tasks where the robot must grasp and hold an object while the object is being perturbed by external forces. Practical examples include bi-manual assembly or disassembly tasks, in which one robot arm grasps an object while another arm applies forces to the object using a tool such as *e.g.* a screwdriver. Fig. 7.1 shows a pair of large robot arms in the nuclear industry, where it is necessary to cut open old nuclear waste containers to examine and safely re-package the contents in modern containers. While teleoperation is currently the most often selected option, there is great interest in applying autonomous control technologies in future. In these tasks, we observe complex manipulations, *e.g.* one arm pushing a large waste drum while another arm cuts it open using hydraulic shears. The motions of pushed objects can be very complex and are difficult to predict and control[MGES^{+}16, SZS20]. This becomes even more complex when an object occupies an unstable equilibrium between opposing pushing forces, *e.g.* between two manipulators.

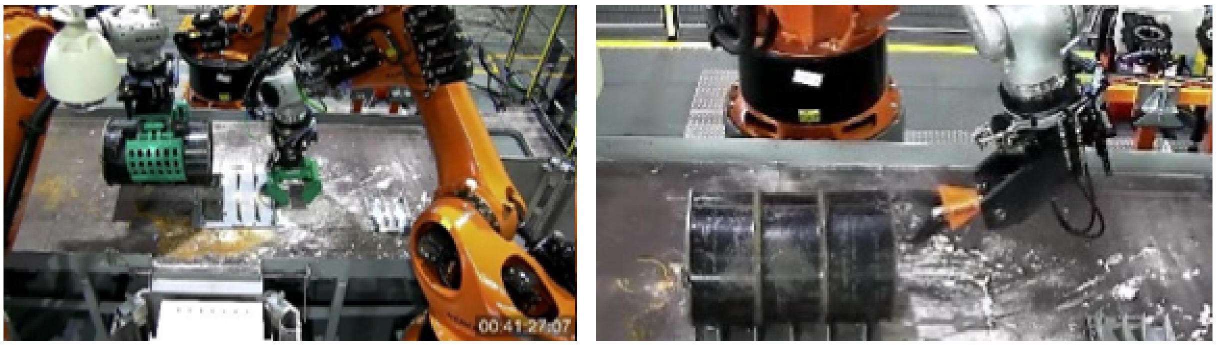


Figure 7.1: Two operators teleoperate two industrial robotic arms to move the barrel at the centre of the bench and cut it open using hydraulic shears.

We can frame such dual-arm problems in terms of a supporting arm and a tooling arm. The tooling arm interacts with the object and applies a profile of forces onto the object's surface (external actions). The trajectory of the tooling arm might be entirely planned, *e.g.* in autonomous robotic cutting [POF$^+$20b]. In the picture, we represent tooling arm as robotic arm, however, we stress that the presence of a second manipulator is not mandatory as long as we have a second actor exerting the forces. In the experimental section, we envisage a human-robot interaction scenario where the supporting arm is autonomous and tasked with stabilising an object against perturbations from a teleoperated tooling arm.

The supporting arm needs to select a grasp on the object suitable for holding the object at the desired configuration, despite the forces generated by the dynamics of the object (internal actions).

Work such as [MGES$^+$16] considers these internal actions for selecting the grasp yielding the minimum robot torque efforts during manipulative movements. Other works investigate the same problem based on external actions. [CFD18] divides a task into a sequence of forceful actions and chooses the grasping point maximising the success of each sub-task. However, these works assume to have a secure grasp of the object throughout the interaction, and they do not generalise to different force profiles or motions. Moreover, we often have imperfect models of robots, objects, and external interactions. Thus, undesired slips of the grasped object may occur during the post-grasp manoeuvre.

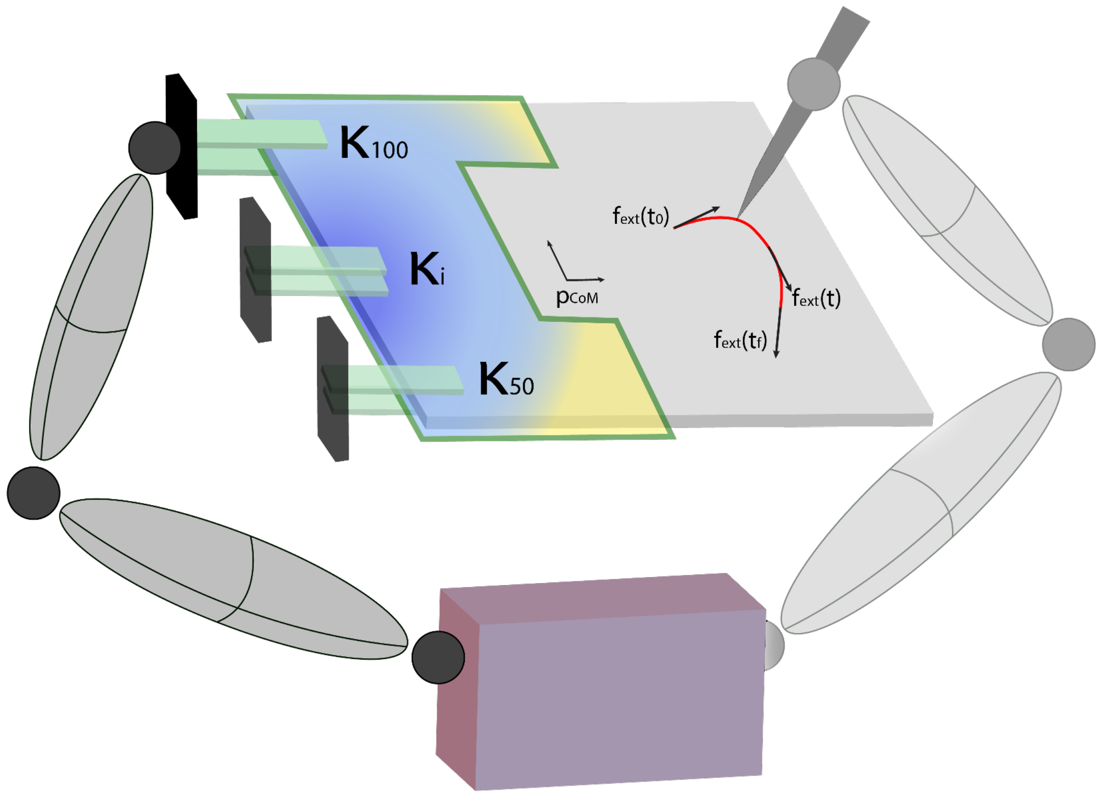


Figure 7.2: The tooling arm (right manipulator) exerts the force profile (in red) onto the object. The supporting arm (left manipulator) must select the best grasping point among 150 grasping candidates sampled within the green boundary. A gradient shows the objective function for every location based on eq. (7.9), low values are shown in blue and high values in yellow, and the three green jaw grippers report proposed grasp candidates on the object.

We may consider the connection between object and gripper as a passive joint with the friction driving the behaviour: high friction, the object never drifts after the grasping; no friction, the passive joint has complete freedom to move; low friction, the connection is loose, *i.e.*, the joint moves, but only under certain forces.

At high friction, the connection is solid, and the controller must govern the system with the arbitrary inertia attached to the end-effector. Works such as [Tho99, ML04, PYPY18b], propose the use of Variable Structure Control , to robustly stabilise the system and solve either the *point-tracking* or the *trajectory-tracking* problem with uncertainties. Additionally, works such as [CK89] propose adaptive force control solutions.

At no friction, the system behaves like a robot with n active joints plus one passive joint. Many authors [Spo96, LIMO03, DHW$^+$17] have studied this group of systems exploiting both canonical and robust controllers for bringing the arm around an equilibrium point. However, the control of these systems is still challenging, and we tailor solutions for the specific application at hand. Moreover, the control of a $(n+1)$-system has been proved non-Small Time Local Controllable (STLC) when the gravity term is missing [LIMO03]. Although [DMO00] propose an iterative steering method that brings the system to equilibrium with a double step algorithm, it has the shortcoming of not being capable of promptly answering external forces which may perturb the object, *e.g.*, the operation from the tooling arm. In this chapter, we tackle the case of low friction. Here, the passive joint rotates around its axis only if the torque exerted exceeds the static friction. Works, such as [WWM$^+$19], study how to detect the slip using sensors, and others exploit this concept to make in-hand manipulation by pivoting [VKSK16, CS17, HJJM16]. Generally, they modulate the grasping force on the object (normal to the grasping point) and leverage the gravitational pull for rotating the object to the desired pose. While the former group deals with the slip after it happened, the latter assumes to have enough grip between the object and gripper to hold it in position. However, this assumption may not hold if we pick up something with unknown properties (resulting in internal actions) or the tooling arm interacts with the object (external actions). Therefore, we propose a strategy to avoid the slip of the grasped object, which accounts for both external and internal forces. A novel grasp selection algorithm minimises the strain on the actuators provided a post-grasp external force profile. Additionally, we propose a robust adaptive controller that accounts for the maximum torque applied to the $(n+1)$ joints and deals with uncertainties in the model. Oppositely to works like [KG13], which designs a controller for cancelling out the dry friction on the actuators, our approach is to exploit the static friction as a constraint, such that the contact between gripper and object never slips.

Contribution

This chapter tackles the discussed shortcomings and proposes an algorithm to select the grasping candidate with the least possibility of letting the object slip for a given external force profile. Moreover, we introduce a control strategy to avoid slippage of the object during the motion of the supporting arm.

The presented novelties are:

- we introduce a novel grasp selection algorithm that accounts for internal and external actions throughout the robot/object interaction by measuring the strain at the joints and the drift of the grasped object from its desired position;
- we propose an integral adaptive sliding mode controller for the supporting arm, which estimates the missing physical terms, rejects *matched* and *unmatched* uncertainties, and guarantees to avoid the slippage of the object;
- we model the system "arm plus object" as composed of n actuated joints plus one passive joint before the end effector. Since the passive joint's static friction poses limits on the admissible torques carried out on the n joints, we introduce a sufficient condition to avoid the slip by solving a minimisation problem.
- we demonstrate the effectiveness of our controller and the novel grasp selection strategy on a simulated (2 + 1)-joint planar robot throughout an extensive set of experiments and analyses, and we also carry out an experiment with a 7 dof Panda robot forced to execute only planar movements.

7.2 Mathematical formulation

We address the problem of an asynchronous dual-manipulation, where one arm (tooling arm) carries out a full-force operation on an object, and the second manipulator grasps and holds the object at an operational configuration. Since we assume have no control over the tooling arm, we must focus on controlling the supporting arm, which holds the object, using a jaw gripper, and opposes internal and external actions. The real dynamics of the object are unknown, and the point of contact allows torsional slip beyond a certain threshold. A static friction estimation is provided using offline identification procedures. We describe the mechanical system with a configuration vector $\boldsymbol{q} \in \mathcal{Q}$, which is a *(n+1)*-dimension vector in the admissible vector space $\mathcal{Q}$. The Euler-Lagrange equations of motions are:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\dot{\boldsymbol{q}}, \boldsymbol{q}) = \boldsymbol{\tau} - \boldsymbol{F}(\dot{\boldsymbol{q}}) + \boldsymbol{d} \tag{7.1}$$

where $\boldsymbol{M}$ is the inertia matrix, $\boldsymbol{h}$ the Coriolis and gravitational pull, $\boldsymbol{\tau}$ the generalised torques applied to the system, $\boldsymbol{F}$ the friction opposing the movement, $\boldsymbol{d}$ is the disturbance on the input, and $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}}$ are respectively the first and second derivatives of $\boldsymbol{q}$[1].

The state of the system could be partitioned in two subsets, $\boldsymbol{q}_a \in \mathcal{Q}_a$ for the n actuated joints and a scalar $q_p \in \mathcal{Q}_p$ for the passive $(n+1)th$ joint, $\boldsymbol{q} = [\boldsymbol{q}_a \; q_p]^T$. Using this partitioning, (7.1) can be written as

$$\begin{aligned} \boldsymbol{M}_{11}\ddot{\boldsymbol{q}}_a + \boldsymbol{M}_{12}\ddot{q}_p + \boldsymbol{h}_a &= \boldsymbol{\tau}_a - \boldsymbol{f}_a + \boldsymbol{d}_a \\ \boldsymbol{M}_{21}\ddot{\boldsymbol{q}}_a + \boldsymbol{M}_{22}\ddot{q}_p + h_p &= -f_p \end{aligned} \tag{7.2}$$

where $\boldsymbol{f}_a$ and f_p are the frictional terms. The generalised torque belonging to the passive joint is zero ($\tau_p = 0$), because no direct action is possible on q_p[2]. The *noncollocated* system in Eq. (7.2) can be expressed using the canonical control form as

$$\begin{cases} \ddot{\boldsymbol{q}}_a = \boldsymbol{B}^{-1}(\boldsymbol{\tau} - \boldsymbol{f}_a + \boldsymbol{d} - \boldsymbol{N} + \boldsymbol{M}_{12}\boldsymbol{M}_{22}^{-1} f_p) \\ \ddot{q}_p = \boldsymbol{M}_{22}^{-1}[-h_p - f_p - \boldsymbol{M}_{21}\ddot{\boldsymbol{q}}_a] \end{cases} \tag{7.3}$$

where $\boldsymbol{B} = \boldsymbol{M}_{11} - \boldsymbol{M}_{12}\boldsymbol{M}_{22}^{-1}\boldsymbol{M}_{21}$, and $\boldsymbol{N} = \boldsymbol{h}_a - \boldsymbol{M}_{12}\boldsymbol{M}_{22}^{-1}(h_p + f_p)$.

[1]We drop the dependencies on $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ for the sake of readability.

[2]In the following text, we drop the suffix a for the torque $\boldsymbol{\tau}_a$ (and therefore $\boldsymbol{d}_a$), as it yields no confusion to the notation.

7.2.1 Static and dynamic friction

The friction term in (7.1) resists to the movement of each joint with a non-linear behaviour, which is proportional and in opposite direction to the joint velocity. A common formulation of the friction between two objects in contact is provided by a combination of two components, (7.4).

$$\boldsymbol{F} = \boldsymbol{F}_s + \boldsymbol{F}_c \tag{7.4}$$

$\boldsymbol{F}_s$ is the static friction (or *stiction*), and $\boldsymbol{F}_c$ models the dynamic friction throughout the sliding. The two regimes act differently on the system. When the torque applied to the robot is below the stiction coefficient, T_s, the torque is cancelled out from an equal but opposite force, as per (7.5).

$$\boldsymbol{F}_s(\boldsymbol{\tau}) = \begin{cases} \boldsymbol{\tau}, & |\boldsymbol{\tau}| \leq T_s \\ 0, & |\boldsymbol{\tau}| > T_s \end{cases} \tag{7.5}$$

Or alternatively, in the more compact form:

$$\boldsymbol{F}_s(\boldsymbol{\tau}) = \frac{\boldsymbol{\tau}}{2}\left[1 - \operatorname{sgn}(|\boldsymbol{\tau}| - T_s)\right] \tag{7.6}$$

As for the dynamic friction, the Coulomb model is widely used, because of its simplicity. It is steered by the sign of the velocity and, it can account for the viscosity friction.

$$\boldsymbol{F}_c = T_c \operatorname{sgn}(\dot{\boldsymbol{q}}) + \beta\dot{\boldsymbol{q}} \tag{7.7}$$

where T_c is the Coulomb coefficient, and β is the viscosity friction coefficient which is steered by the velocity. In this work, we overlook the viscosity behaviours ($\beta = 0$).

Notice that the sgn(·) function is defined as

$$\operatorname{sgn}(x) = \begin{cases} 1, & x > 0 \\ 0, & x = 0 \\ -1, & x < 0 \end{cases} \tag{7.8}$$

7.3 Grasp selection algorithm

Consider a discrete set of grasp candidates, K, for an object, like in Fig. 7.2, and a candidate κ within this set. When the robot picks the object at κ, the newly kinematic chain is described by eq. (7.2). Internal and external actions greatly impact the strain on the robot throughout the task. It is due to potential displacement from the object's Centre of Mass (CoM) and κ and the low friction at the grasp location. A grasp candidate close to the CoM of the object will exert small forces on the passive joint. In contrast, a grasp on a location far from the CoM would create big swings, and the controller would need to put a high control effort to overcome the object dynamics.

Let's assume that an algorithm, *i.e.* [POF+20b], provides a path or a trajectory of forces applied by a tooling arm onto the object to manipulate, $\boldsymbol{f}_{ext}$, and that estimates of $\hat{\boldsymbol{M}}$ and $\hat{\boldsymbol{h}}$ are available. We want to select the grasp candidate that minimises the control effort $\boldsymbol{u}(t)$ throughout the entire execution while being subjected to external and internal actions. We formally describe the grasp selection problem as:

$$\begin{aligned} \kappa^* = \min_{\kappa \in K} & \int_{t_0}^{t_f} \boldsymbol{u}(t)^T \boldsymbol{W}_u \boldsymbol{u}(t) + w_p e_p^2 \, dt \\ & \text{s.t. } \boldsymbol{f}_{ext}(t), \ t \in [t_0, t_f] \end{aligned} \tag{7.9}$$

where $\boldsymbol{f}_{ext}(t)$ is a profile of external forces carried out by the tooling arm onto the object, $[t_0, t_f]$ is the interval of the interaction, κ^* is the optimal grasping candidate, and the argument of the integral is the function to minimise at every instant. The first term, $\boldsymbol{u}^T \boldsymbol{W}_u \boldsymbol{u}$, accounts for the energy used by the system to maintain the object at the operational point and weights the contributions by the matrix $\boldsymbol{W}_u$. The term $w_p e_p^2$ penalises the drift, $e_p = q_p - q_{p|eq}$ and $w_p > 0$. Thanks to the estimated model, we account for the internal actions. We play out the system against the external forces in a simulated scenario, and we measure the torques applied to hold the robot in the desired position. Every grasp configuration generates a different effort, and the recorded values can be evaluated using the objective function in (7.9) to select the best grasp location.

7.4 Robust controller design

The control input $\boldsymbol{u}(t)$ in (7.9) must jointly deal with the uncertainties present in the system and the no-slip requirement at the passive joint. Therefore, it demands a particular design for attaining a set of admissible internal torques. VCS achieve great results in rejecting uncertainties, and a popular solution within the VCS class is Sliding Mode Control (SMC). The design of SMC consists of two stages: the selection of a *sliding surface* (or *hyper-surface*), which imposes arbitrary dynamics on the system; and an initial *reaching phase*, which brings the system to the sliding surface. Once we reach the sliding surface, a non-linear controller forces the system on the sliding surface.

Consider the sub-system of the first n equations in Eq. (7.3) to be affected by both *matched* and *unmatched* uncertainties[1]. Every term can be written using the sum of two elements, a nominal value plus the error on the estimation, such as, $\boldsymbol{B} = \hat{\boldsymbol{B}} + \delta\boldsymbol{B}$ and $\boldsymbol{N} = \hat{\boldsymbol{N}} + \delta\boldsymbol{N}$, and we assume the errors to be bounded, $|\boldsymbol{B}| \leq \boldsymbol{\Gamma}_1$ and $|\boldsymbol{N}| \leq \boldsymbol{\Gamma}_2$. Thus, we write the new uncertain dynamics as:

$$\hat{\boldsymbol{B}} \ddot{\boldsymbol{q}}_a + \hat{\boldsymbol{N}} = \boldsymbol{\tau} - \boldsymbol{\psi} + \boldsymbol{d} \tag{7.10}$$

[1]Given the system $\dot{x} = Ax + bu$, we say the uncertainty d is matched if we can write it as $\dot{x} = Ax + b(u + d)$; unmatched otherwise.

where we lump in $\boldsymbol{\psi}$ all the errors due to the estimation process (therefore, if $\hat{\boldsymbol{B}} \to \boldsymbol{B}$ and $\hat{\boldsymbol{N}} \to \boldsymbol{N}$ then $\boldsymbol{\psi} \to 0$), and $\boldsymbol{d}$ accounts for the disturbances, with $|\boldsymbol{d}| \leq \boldsymbol{\Delta}$. In (7.10), we consider the friction on the actuated joints, $\boldsymbol{f}_a$, a disturbance.

7.4.1 Integral adaptive sliding mode control

We want to design a controller that stabilises the robot around an equilibrium point, $\boldsymbol{x}_{eq} \in \mathcal{Q}^{n\times 3}$, under the effect of *matched* and *unmatched* disturbances. We propose to use an Integral Adaptive Sliding Mode Control (I-ASMC), which brings the system to the desired equilibrium while estimating the model error $\hat{\boldsymbol{\psi}}$, with an observer. The SMC naturally tackles *matched* uncertainties (and by extension, ASMC does as well) but falls short in attaining stability during the reaching phase if the system is affected by *unmatched* uncertainties. Therefore, we propose to use an integral term that removes this initial phase and forces the system to start on the sliding surface.

Consider an equilibrium point $\boldsymbol{x}_{eq} = (\boldsymbol{q}_{eq},\ \dot{\boldsymbol{q}}_{eq},\ \ddot{\boldsymbol{q}}_{eq})$ and a sliding surface:

$$\boldsymbol{s} = \dot{\boldsymbol{e}} + k_1\boldsymbol{e} + k_2\int^t \boldsymbol{e}dr + k_2 e(t_0) \tag{7.11}$$

where $\boldsymbol{e} = \boldsymbol{q} - \boldsymbol{q}_{eq}$ is the configuration error, $\dot{\boldsymbol{e}}$ is the first derivative of $\boldsymbol{e}$, and $\boldsymbol{e}(t_0)$ is value of the configuration error at the initial time t_0. The control input:

$$\boldsymbol{u} = \hat{\boldsymbol{N}} + \hat{\boldsymbol{B}}(\ddot{\boldsymbol{q}}_{eq} - k_1\dot{\boldsymbol{e}} - k_2\boldsymbol{e}) + \boldsymbol{u}_s \tag{7.12}$$

stabilises the system (7.10) around $\boldsymbol{x}_{eq}$, when the switching control $\boldsymbol{u}_s$ is:

$$\boldsymbol{u}_s = \hat{\boldsymbol{\psi}} - \hat{\boldsymbol{B}}(\eta\ \mathrm{sgn}(\boldsymbol{s}) + \lambda\boldsymbol{s}) \tag{7.13}$$

An observer for $\boldsymbol{\psi}$ is designed as:

$$\dot{\hat{\boldsymbol{\psi}}} = -\gamma\hat{\boldsymbol{B}}^{-T}\boldsymbol{s} \tag{7.14}$$

and if $\gamma > 0$, k_1 and k_2 are Hurwitz, $\lambda > 0$, and $\eta > |\boldsymbol{\Gamma}_1^{-1}||\boldsymbol{\Delta}|$.

Proof. We use the direct Lyapunov method to demonstrate the stability of the system. Consider the function:

$$V = \frac{1}{2}\boldsymbol{s}^T\boldsymbol{s} + \frac{1}{2\gamma}\tilde{\boldsymbol{\psi}}^T\tilde{\boldsymbol{\psi}}, \tag{7.15}$$

such function is $V > 0$ and $V(0) = 0$. $\tilde{\boldsymbol{\psi}} = \hat{\boldsymbol{\psi}} - \boldsymbol{\psi}$ is the estimation error on the unmodelled dynamics. We need to prove that $\dot{V} < 0$.

$$\dot{V} = \boldsymbol{s}^T\dot{\boldsymbol{s}} + \frac{1}{\gamma}\dot{\tilde{\boldsymbol{\psi}}}^T\tilde{\boldsymbol{\psi}} \tag{7.16}$$

Although the component $\boldsymbol{\psi}$ is dynamic, we can assume its derivative to change slower than its estimation, which yields the approximation $\dot{\tilde{\boldsymbol{\psi}}} = \dot{\hat{\boldsymbol{\psi}}} - \dot{\boldsymbol{\psi}} = \dot{\hat{\boldsymbol{\psi}}}$:

$$\dot{V} = \boldsymbol{s}^T(\ddot{\boldsymbol{e}} + k_1\dot{\boldsymbol{e}} + k_2\boldsymbol{e}) + \frac{1}{\gamma}(-\gamma\hat{\boldsymbol{B}}^{-T}\boldsymbol{s})^T\tilde{\boldsymbol{\psi}} \tag{7.17}$$

using eq. (7.10):

$$\dot{V} = \boldsymbol{s}^T[\hat{\boldsymbol{B}}^{-1}(\boldsymbol{\tau} - \boldsymbol{\psi} - \hat{\boldsymbol{N}} + \boldsymbol{d}) - \ddot{\boldsymbol{q}}_{eq} + k_1\dot{\boldsymbol{e}} + k_2\boldsymbol{e}] - \boldsymbol{s}^T\hat{\boldsymbol{B}}^{-1}\tilde{\boldsymbol{\psi}} \tag{7.18}$$

Controlling $\boldsymbol{\tau}$ with the proposed $\boldsymbol{u}$, we obtain:

$$\begin{aligned}
\dot{V} &= \boldsymbol{s}^T\{\hat{\boldsymbol{B}}^{-1}[-\boldsymbol{\psi} + \boldsymbol{d} + \boldsymbol{u}_s]\} - \boldsymbol{s}^T\hat{\boldsymbol{B}}^{-1}\tilde{\boldsymbol{\psi}} \\
&= \boldsymbol{s}^T\{\hat{\boldsymbol{B}}^{-1}\boldsymbol{d} - \hat{\boldsymbol{B}}^{-1}[(\boldsymbol{\psi} + \tilde{\boldsymbol{\psi}}) - \boldsymbol{u}_s]\} \\
&= \boldsymbol{s}^T\{\hat{\boldsymbol{B}}^{-1}\boldsymbol{d} - \hat{\boldsymbol{B}}^{-1}[\hat{\boldsymbol{\psi}} - \boldsymbol{u}_s]\} \\
&= \boldsymbol{s}^T\{\hat{\boldsymbol{B}}^{-1}\boldsymbol{d} - \hat{\boldsymbol{B}}^{-1}[\hat{\boldsymbol{\psi}} - \hat{\boldsymbol{\psi}} + \hat{\boldsymbol{B}}(\eta\text{sgn}(\boldsymbol{s}) + \lambda\boldsymbol{s})]\}
\end{aligned} \tag{7.19}$$

and after the simplification:

$$\dot{V} = \boldsymbol{s}^T\{\hat{\boldsymbol{B}}^{-1}\boldsymbol{d} - \eta\text{sgn}(\boldsymbol{s}) - \lambda\boldsymbol{s}\} \tag{7.20}$$

If $\lambda > 0$ and $\eta > 0$, the second and third terms are negative[1], but nothing can be said about the first term in this formulation, therefore:

$$\dot{V} \leq \boldsymbol{s}^T\{|\boldsymbol{\Gamma}_1^{-1}||\boldsymbol{\Delta}| - \eta \text{sgn}(\boldsymbol{s}) - \lambda \boldsymbol{s}\} \tag{7.21}$$

When the gain $\eta > |\boldsymbol{\Gamma}_1^{-1}||\boldsymbol{\Delta}|$, the first term is overstepped by the non linear contribution and $\dot{V} < 0$, which guarantees the asymptotic stability of the system using the non linear controller (7.12).

□

As in [Liu17], we bound the inputs $\hat{\boldsymbol{\psi}}$ to avoid values too high in the control law. We substitute $\hat{\boldsymbol{\psi}}$ with $sat(\hat{\boldsymbol{\psi}})$, which bounds the value between $[\boldsymbol{\psi}_{min}, \boldsymbol{\psi}_{max}]$.

We define the saturation function as:

$$sat(x) = \begin{cases} x_{max}, & x > x_{max} \\ x_{min}, & x < x_{min} \\ x, & otherwise \end{cases} \tag{7.22}$$

When the system is on the sliding surface $\{\boldsymbol{s}, \dot{\boldsymbol{s}}\} = 0$, then:

$$\ddot{\boldsymbol{e}} + k_1\dot{\boldsymbol{e}} + k_2\boldsymbol{e} = \boldsymbol{0} \tag{7.23}$$

where k_1 and k_2 are positive definite and Hurwitz. Because of the uncertain dynamics of the system, a natural method for tuning the gains is using trial-and-error until we reach a tuple that satisfies the boundaries of the friction at the passive joint. However, the tuning is time-consuming and not always practical, *i.e.* all grasped objects must go through the tuning phase or the movements are constrained by a cluttered environment. We, therefore, propose an approach to select a pair (k_1, k_2) that allows no slip of the object while the robot is moving.

[1]We notice that the $s^T\text{sgn}(s)$ can be written as $(s^T s)/|s|$

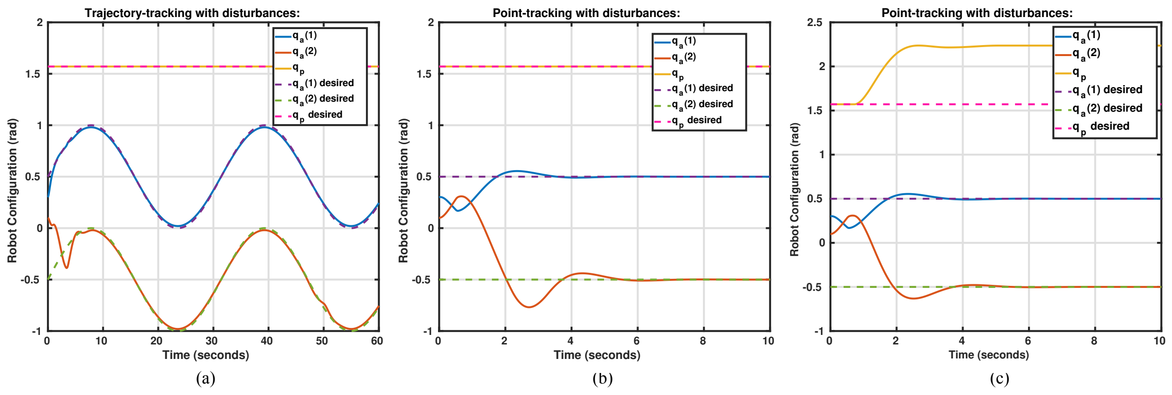


Figure 7.3: We report the response of our controller on the *trajectory-tracking* and *point-tracking* problem on a (2 + 1) joints robot. In Fig. 7.4.1a), the robot must follow a sinusoidal movements while avoid slip of the passive joint. The control coefficients comply with the inequality (7.28), and the stiction at the passive joint is 5 N. After less than 10 seconds, the robot reaches the trajectory with an error smaller than 0.03 rads thereafter. Fig. 7.4.1b) and Fig. 7.4.1c) report the *point-tracking* of an operational configuration. In the former, the stiction is 5 N and the controller gains are the same as the *trajectory-tracking* case. In the latter, we consider static friction of 3 N, but we do not update the coefficients according to the method proposed in Sec. 7.4.1. Because of the lower friction, the maximum torque for each actuator is reduced but not accounted for it. Thus, the object slips, yellow solid line at the top of the picture. The dashed lines show the reference trajectories for the joints, and the solid lines depict the output of the system. The commanded torques are affected by a disturbance $\boldsymbol{d} = [0.2sin(0.5t), 0.2sin(0.5t)]$.

7.4.2 Robust gains tuning

First, we find the maximum torques for the actuated joints, and we set up a minimisation problem to stay within the boundaries of the estimated stiction, T_s. Second, we show how the constraint of the inputs yields a sufficient condition for the pole-placement technique.

Consider the system in eq. (7.1), when the torque at the passive joint is less than the static friction, $\dot{q}_p = 0, \ddot{q}_p = 0$, we then obtain (after a simple manipulation):

$$\boldsymbol{M}_{21}\boldsymbol{M}_{11}^{-1}\boldsymbol{\tau} = \tau_p + \boldsymbol{M}_{21}\boldsymbol{M}_{11}^{-1}(\boldsymbol{h}_a + \boldsymbol{f}_a) - h_p \tag{7.24}$$

where τ_p is the torque exerted to the passive joint and fully cancelled out by the static friction.

To obtain the no-slip condition of the (n+1)th joint, the constraint $|\tau_p| \leq T_s$ must always be satisfied. Since an accurate model is not available in many cases, we resort to an estimation of the system, and we assume the error between the real and estimated model is bounded for every t, with $t > t_0$ and t_0 the initial time. The lower upper bounds provide a set of inequalities that may be reshaped by the mean of a matrix $\boldsymbol{A}_i$ and a vector $\boldsymbol{b}_i$ which constrains the $\boldsymbol{\tau}$:

$$\boldsymbol{A}_i\boldsymbol{\tau} \leq T_s + \boldsymbol{b}_i \tag{7.25}$$

The matrix $\boldsymbol{A}_i$ is a $2k \times n$ matrix, where k is the number of combinations for the upper an lower bounds, and $\boldsymbol{b}_i$ is a $2k \times 1$ vector. The first k set of inequalities accounts for the case $\tau_p \leq T_s$, while the second set accounts for $\tau_p \geq -T_s$.

We want to find the maximum torques that satisfy the inequalities. We therefore solve the dual minimisation problem such as:

$$\begin{aligned} \min_{\tau} & -\boldsymbol{\tau}^T\boldsymbol{W}\boldsymbol{\tau} \\ \text{s.t. } & \boldsymbol{A}_i\boldsymbol{\tau} \leq T_s + \boldsymbol{b}_i \\ & \boldsymbol{\tau} \leq \boldsymbol{\tau}_M \\ & -\boldsymbol{\tau} \leq -\boldsymbol{\tau}_m \end{aligned} \tag{7.26}$$

where $\boldsymbol{\tau}_M$ and $\boldsymbol{\tau}_m$ are maximum and minimum feasible torques for the actuated joints, and $\boldsymbol{W}$ is the matrix weighting the actuators effort. If we prove that the $(2k + 2n)$ set of inequalities define a convex region, and the problem admits a global minimum, the solution of (7.26) always attains a vector of maximum torques for the uncertain system at hand.

As for the convexity, it is trivial to show that all the constraints are planes (or hyperplanes), which are convex sets, and we know that the intersection of any convex set is convex. As for the global minimum, we look at Eq. (7.26). The objective function is the negation of the weighted square of the norm of the torques' vector, which has a maximum in zero and a global minimum on one of the active constraints. Since the objective function is concave, the minimisation takes longer than a quadratic programming problem, but it is trivial to solve offline with one of the many non-linear optimisation toolboxes in literature. We care to notice that the given solution is only sufficient, and a better solution might exist. Although our approach is conservative, it assures that the torque at the passive joint never oversteps the static friction.

Now that we have obtained the vector of maximum torques, $\bar{\boldsymbol{\tau}}$, we want to find a suitable Hurwitz set of parameters (k_1, k_2). Given the control input in eq. (7.12), we write:

$$\begin{aligned} |\boldsymbol{u}| &= |\hat{\boldsymbol{N}} + \hat{\boldsymbol{B}}(\ddot{\boldsymbol{q}}_{eq} - k_1\dot{\boldsymbol{e}} - k_2\boldsymbol{e}) + \boldsymbol{u}_s| \\ &\leq |\hat{\boldsymbol{N}}| + |\hat{\boldsymbol{B}}(\ddot{\boldsymbol{q}}_{eq} - k_1\dot{\boldsymbol{e}} - k_2\boldsymbol{e})| + |\boldsymbol{u}_s| \\ &\leq |\hat{\boldsymbol{N}}| + |\hat{\boldsymbol{B}}|(|\ddot{\boldsymbol{q}}_{eq}| + k_1|\dot{\boldsymbol{e}}| + k_2|\boldsymbol{e}|) + |\boldsymbol{u}_s| \\ &\leq \boldsymbol{\Gamma}_2 + \boldsymbol{\Gamma}_1(|\ddot{\boldsymbol{q}}_{eq}| + k_1|\dot{\boldsymbol{e}}| + k_2|\boldsymbol{e}|) + |\boldsymbol{u}_s| \end{aligned} \tag{7.27}$$

where we have overestimated the control input $|\boldsymbol{u}|$ using the known bounds on the estimated values. In real applications, the errors are usually bounded by the actuators. Hence, using the saturation on the errors and the maximum torque computed by (7.26), we obtain:

$$|\bar{\boldsymbol{\tau}}| \geq \boldsymbol{\Gamma}_2 + \boldsymbol{\Gamma}_1(sat(\ddot{\boldsymbol{q}}_{eq}) + k_1 sat(\dot{\boldsymbol{e}}) + k_2 sat(\boldsymbol{e})) \tag{7.28}$$

which expresses a constraint for the selection of the gains (k_1, k_2).

7.5 Experimental results

We demonstrate the effectiveness of the introduced grasp selection algorithm and the I-ASMC controller in the challenging case of a planar robot. We report an extensive analysis of the controller behaviour and the grasp selection algorithm on a simulated (2 + 1) planar robot. We initially show the performance of the controller against internal actions, where the tracking of the reference trajectory task is affected by *matched* and *unmatched* uncertainties. Furthermore, a set of experiments is designed to display the necessity for the selection of suitable controller gains so that we guarantee no slip between gripper and object. We carefully analyse the grasp selection algorithm over different CoM positions and external actions. Here, we assume an estimation of the object's dynamics to be provided. Additionally, we test the proposed pipeline (grasp selection + controller) on a real 7 dof Panda robot. We constrain the robot to only planar movements actuating only the first and sixth joints.

7.5.1 I-ASMC

We consider the system is affected by a disturbance acting on the torque inputs as $\boldsymbol{d} = [0.2sin(0.5t), 0.2sin(0.5t)]$, and the inertia of the grasped object is unknown. The disturbance is unknown to the controller, and only the boundaries are provided. The coefficients of the controller are selected following the rule in Sec. 7.4.1. We task the arm to follow a reference trajectory, $\boldsymbol{r}(t) = [0.5\ sin(0.2t), 0.5\ sin(0.2t)]$, whose frequency is less than half of the frequency of the disturbing signal.

In Fig. 7.4.1a), we compare the output of the system (solid lines) against the reference trajectories (dashed lines). The actuated joints (blue and red lines) reach the convergence to the desired values after less than 10 seconds and with an error in magnitude less than 0.03 rads thereafter. The yellow line in Fig. 7.4.1a) show that the torque applied to the passive joint is always below the stiction (5 N in this experiment).

We also test our controller on the *point-tracking* problem. The cases of high (above 10 N) and low friction (below 3 N) for the passive joint are shown using Figs. 7.4.1b) and 7.4.1c) respectively. Although both the experiments show a robust behaviour for the dynamics of the object to the actuated joints, the low friction case shows that the selection of the controller coefficients is not suitable for avoiding the slip. In both experiments, the transient state lasts less than 6.5 seconds and overshoots the desired values. In the low friction experiment, the swing of the object creates an additional dynamic behaviour that

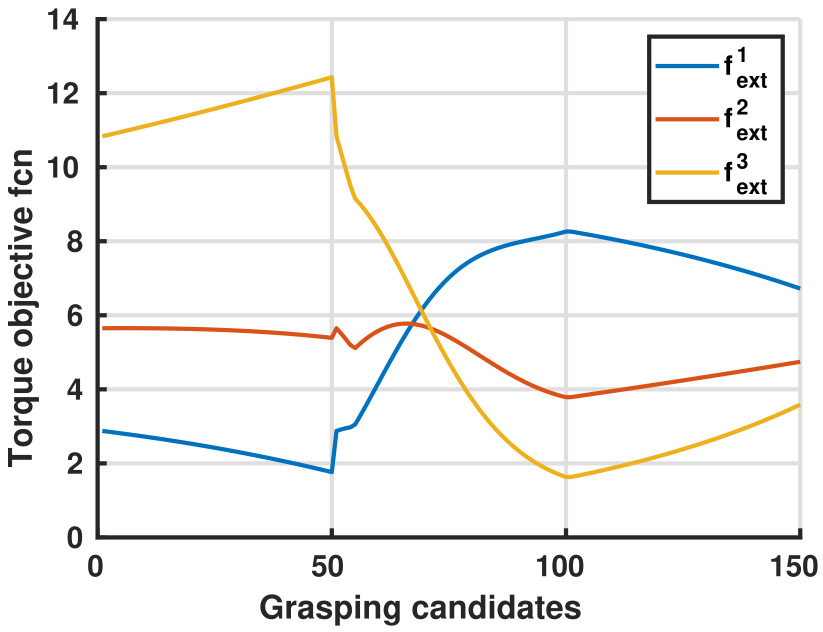
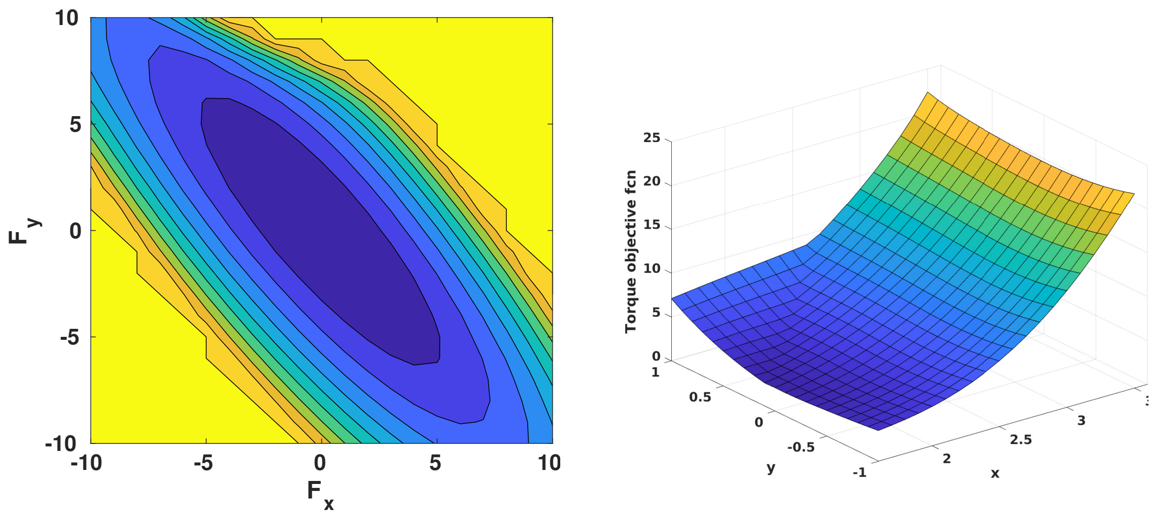


Figure 7.4: We report the control effort for 150 grasping candidates exposed to three force profiles, $\boldsymbol{f}^1_{ext} = [1\ 0\ 0]^T$, $\boldsymbol{f}^2_{ext} = [0\ 0.5\ 0]^T$, $\boldsymbol{f}^3_{ext} = [0\ 0\ 1]^T$. The first 50 locations are the bottom left of the object, the 50 in the middle are on the left area, and the last 50 are at the top of square.

reduces the convergence time. Since the object swing with opposite direction w.r.t q'_a, it creates a damping effect on the signal overshoot (the lower point of the red line is -0.765 in Fig. 7.4.1b) against -0.5843 in Fig. 7.4.1c().

7.5.2 Robust grasping of an object

We design a set of experiments to show that the grasping location of an object affects the strain on the actuators. We consider a 2-joint robot, which must pick a grey squared object, like the one in picture 7.2. The zone surrounded by the green lines shows the area of reachable grasps for the arm. The nature of the task simplifies the generation of possible grasping candidates that work in accordance with the type of gripper, task, and object. Hence, we decided to hold the gripper perpendicular to the object's edges and create a grid of manually selected points within the area. Then, we interpolated the points with a resolution such that we obtained 50 grasping candidates for each side. At the end of the process, we collected 150 grasping candidates. An inverse kinematic



(a) (b)

Figure 7.5: Fig. 7.5(a) shows the control effort to vary the external forces. We pick a grid of possible forces, $\{F_x, F_y\} = \{[-10, 10], [-10, 10]\}$, fix the CoM at $p_{CoM} = [2.5\ ,0]^T$, and report in yellow the external forces that more excite the system, and in blue the ones with less impact. Fig. 7.5(a) shows the control effort to vary the position of the CoM of the object. We use a grid $\{x, y\} = \{[1.5, 3.5], [-1, 1]\}$ for the p_{CoM} and a the force f^1_{ext}. Yellow areas depicts higher control efforts and dark blue ones lowers. Both experiments fix the grasp point at $\kappa = [1.5\ 0]^T$.

algorithm computes the matching joint configurations.

We assume an estimation of the CoM of the object, p_{CoM} in Fig. 7.2, and a profile of forces acting on the object provided as input. We want to assess the actuator's effort to hold the object at a given equilibrium.

The coupling between grasping point, CoM, and external forces change greatly the strain for the actuators. In picture Fig. 7.4, we report the effort exerted by the actuators for the 150 candidates with three profile of forces applied to the object, $\boldsymbol{f}^1_{ext} = [1\ 0\ 0]^T N$, $\boldsymbol{f}^2_{ext} = [0\ 0.5\ 0]^T$ N, $\boldsymbol{f}^3_{ext} = [0\ 0\ 1]^T N$, and the CoM at $p_{CoM} = [2.5\ ,0]^T$. As explained in Sec. 7.3, a matrix of weights $\boldsymbol{W}_u$ and the coefficient w_p can be used to scale the contribution of each joint, *e.g* a possible choice for the former is to use of the inertia matrix of the robot when available. In our experiments, we employ the constant matrix $\boldsymbol{W}_u = diag(1, 0.7)$, which accounts more for the energy of the first joint than the second, and a penalising factor of $w_p = 10$.

Analysing Fig. 7.4, we clearly see that a minimum exists over the possible grasping locations in all three cases. The first 50 candidates are located at the bottom of the square, 50 are in the middle on the left edge, and the remaining are at the top of the object. As we expect, the best grasping points hugely differ between $\boldsymbol{f}^1_{ext}$ and $\boldsymbol{f}^3_{ext}$, where the force profiles are perpendicular to one another. The best candidates for $\boldsymbol{f}^1_{ext}$ is at the bottom left corner, and represents the worst case for the case $\boldsymbol{f}^3_{ext}$. On the other hand, the best candidate for $\boldsymbol{f}^3_{ext}$ is in the upper left corner, which represents the greater effort against $\boldsymbol{f}^1_{ext}$. As for $\boldsymbol{f}^2_{ext}$, we split the description into two factors: magnitude and profile. In this case, the control effort behaviour mixes some of the results shown from the other two force profiles. It follows $\boldsymbol{f}^1_{ext}$ for the first 50 candidates and then swaps to $\boldsymbol{f}^3_{ext}$ for the candidates on the left and top edges. The magnitude, however, is homogeneous across all candidates with respect to the other experiments; with a maximum span of 1.995, against 6.5 for the first and 10.82 for the third. This result leads our study towards the relation of different force profiles and the point of application p_{CoM} over every grasping point.

We therefore show this behaviour with a second set of experiments, where we pick the grasping candidate $\kappa = [1.5\ 0]^T$, and we span the CoM position over the grid $\{x, y\} = \{[1.5, 3.5], [-1, 1]\}$, Fig. 7.5(b). As we expect, CoMs far from the picking point (yellow colours) make the object swing, which results in a greater control effort for the actuators. On the contrary, grasping the object close to the centre of mass (dark blue) drastically reduces the control effort of the 88%; from around 22.01 in the worst case, to almost 2.64 at the minimum point. We furthermore study the effect of the force profile on the selection process in Fig. 7.5(a). We again chose $\kappa = [1.5\ 0]^T$ and $p_{CoM} = [2.5\ , 0]^T$, but we span the external forces to be $\{F_x, F_y\} = \{[-10, 10], [-10, 10]\}$. The graph shows a trivial minimum in $\{F_x, F_y\} = \{0, 0\}$, but it also highlights the worst-case scenario when the two components of the force are pulling the object in the same direction $\{-10, -10\}$ and $\{10, 10\}$. On the contrary, the control effort stays low when the $\{F_x, F_y\}$ have inverse polarity. In this situation, the vector of external forces is almost in line with the grasping candidate, and the robot requires less energy to react to the disturbances. Generally, if the force profile and the grasping point align, the external forces are passively compensated by the robot; *e.g.* pulling an object grasped by the robot along the x-axis when the arms are in configuration $\boldsymbol{q}_a = [0\ ,0]^T$, results in no actions by the controller.

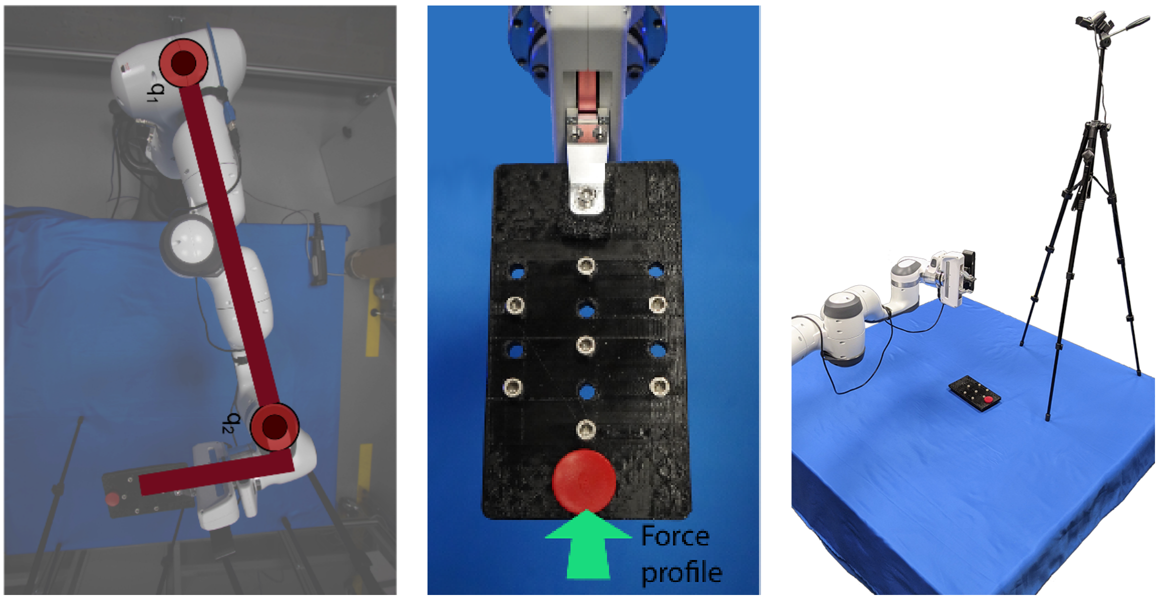
7.5.3 Real robot experiment

We set up an experiment to assess the performance of our I-ASMC, and the grasping algorithm on a real robot provided a task for the tooling arm. We employ a 7 dof Panda robot, but we constrain its movements to planar motions, where we actuate only the first and sixth joints, Fig. 7.6(a). As we explained, this condition makes the $(n + 1)$ system non-STLC. The robot must grasp the object in Fig. 7.6(b), reach the desired location $q_d = [0.3, 1.5675]$ and react to the external forces provided by a human operator who wants to drive a screw in the object. The force has orientation as in Fig. 7.6(b). The object is a 3-D printed rectangular flat plate with 13 holes placed at the centre of the body. Every opening lodges a screw (up to ø7 mm), and we can easily change the position of the CoM and weight of the object by rearranging the screws on the plate.

Our grasping selection algorithm takes as input the previous information and evaluates 150 grasping candidates on the upper side of the plate, similar to the simulation experiment, Fig. 7.7. It yields the best grasp being the one in Fig. 7.6(b) with a minimum value of 5.785.

To enforce the assumption of only torsional slip between object and gripper, we pierce the plate at the grasping location and plug a metal rod which links the two devices. Moreover, we pad the fingers with a smooth fabric that allows low friction letting the object drift. The controller is naive to the real CoM of the object and the inertia properties of the robot, and only an estimate is available. As we described in Sec. 7.4.1, the controller gains play a critical role to attain no-slip of the object while the robot is in motion. Following the result in (7.28), we select $k_1 = 5$ and $k_2 = 30$ as coefficients.

A 2D camera records the motion of a red circle attached to the plate using a colour recognition algorithm written in OpenCV. As Fig. 7.8 shows, the resulting drift between



(a) (b) (c)

Figure 7.6: 7.6(a). Top view of the Panda robot at the configuration $q = [0.5, 1.57, -1.57, -0.08, 0.0, 1.8675, 0.7853]$. We control only the first and sixth joints (red circles with the darker red dot). 7.6(b) depicts the robot grasping the object from the grasping location suggested by our algorithm. The red circle is used for detecting the drift of the object while the robot is moving. An operator apply a force with a screwdriver onto the object, which is directed as the green arrow. 7.6(c) shows the setup of the robot throughout the experiment.

the robot and object, q_p is always less than 0.005 rads. Because of the human task and the low friction, the slip angle rises over 0.03 throughout the interaction. The remaining error is due to the poor frame rate of the camera, which adds noise to the circle track. The initial errors between desired and current configuration are 0.3 and 0.2 respectively for $q_a(1)$ and $q_a(2)$. Fig. 7.8 reports that the actuated joints reach the desired location in less than 6.03 seconds. From the instant 8 to 10 in Fig. 7.8, the operator starts to task. The high-frequency controller promptly rejects the external disturbances and maintains the operational configuration with no relevant change in the actuated configuration.

In Fig. 7.9, we report the task of driving a screw into a plate. The operator uses the screwdriver to apply a force onto the plate edge. In the first row, the robot uses the grasp candidate yielded by our approach. The robot configuration naturally opposes the

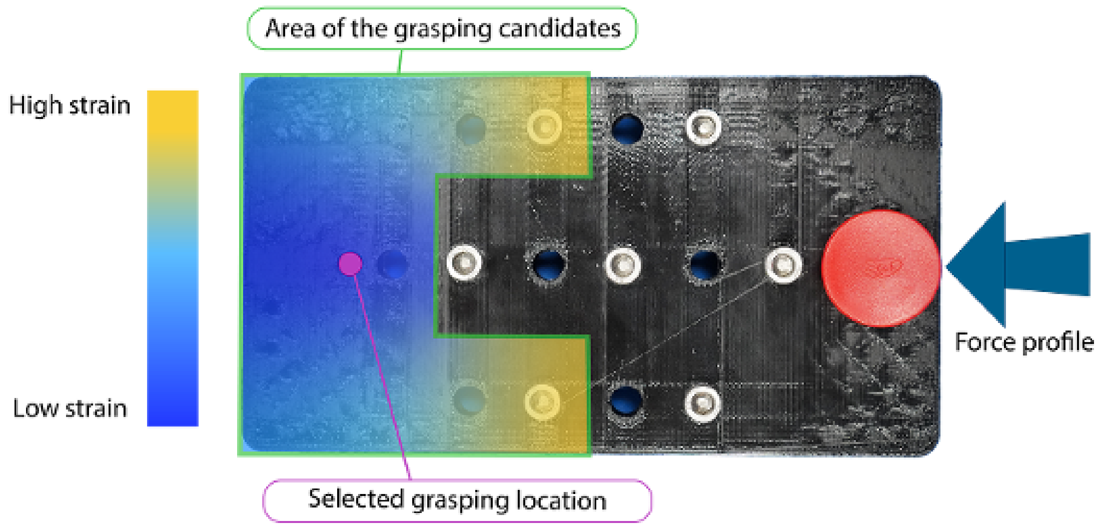


Figure 7.7: We consider a set of 150 grasping candidates on the left side of the object. High strains for the robot's joints are coloured using yellow, whereas low strains are coloured using dark blue. Because of the task (blue arrow), the best grasp falls into the centre of the plate with orientation aligned to the longer side of the object.

operator's force throughout the driving of the screw, and the plate stays in place during the execution.

On the other hand, the bottom row shows the performance of a random grasp candidate. Although the grasp is stable, the operator's actions make the plate drift. Therefore, the operator must re-orient the motion to follow the plate with the task becoming less efficient. After the initial slippage, the object's side starts touching the gripper's body, and the operator can finish the task. However, this is only by chance, and we cannot assure operations to be carried out in all situations.

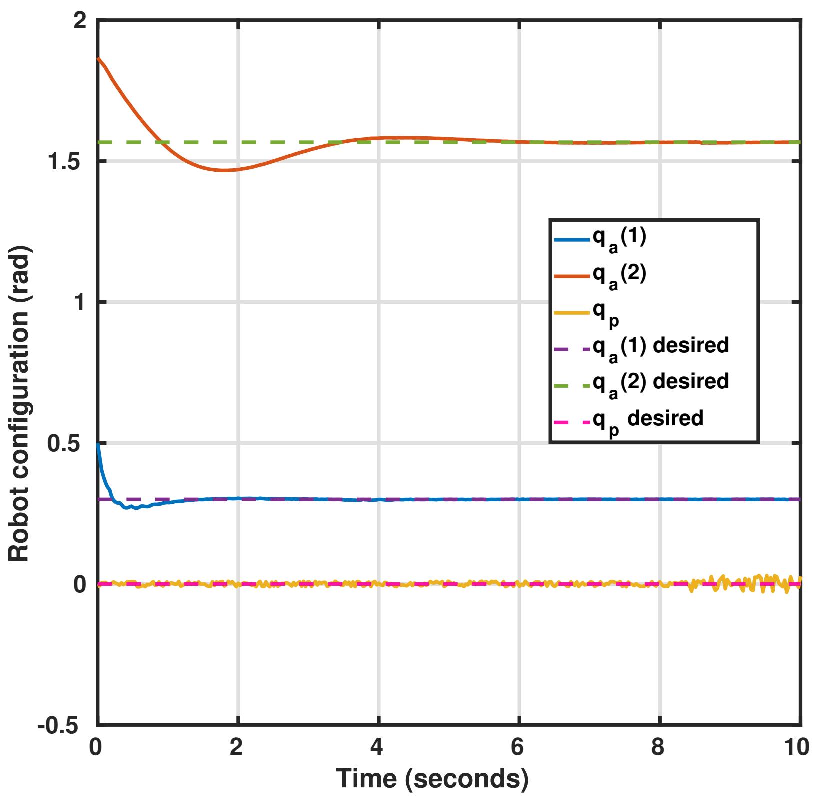


Figure 7.8: We show the movements of the robotic arm. The tracking error becomes less than $3e^{-3}$ after 6.03 seconds.

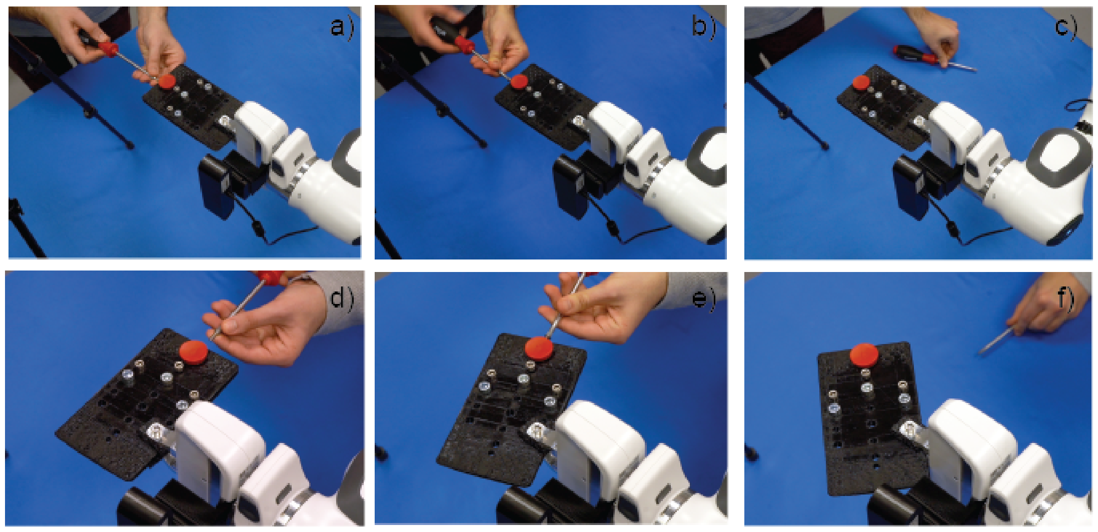


Figure 7.9: First row: the robot selects the grasp candidate yielded by our approach. The operator lodges the screw into its bed, a). Applies pressure to drive the screw, b). And finally, it finishes the task successfully. Second row: the robot selects a random grasp candidate. The operator lodges the screw into its bed, d). While the user applies pressure, the object starts tilting in-hand (slippage), e). The object reaches a stable position by colliding with the gripper's body, and the task has been completed, f).

7.6 Conclusion

In this chapter, we address the problem of dual-arm manipulation of an object with arbitrary dynamics and subject to external actions. One of the arms must grasp the object and hold it to the desired location. Because of the low friction between gripper and object, a rotational slip is allowed depending on the applied torque. We propose to mitigate the action of internal and external forces by carefully selecting the grasping location and the control of the manipulator. We introduce a grasp selection algorithm that accounts for the external forces exerted on the object and yields the minimum control effort for maintaining the robot at the prescribed operational configuration without slip. Our proposed integral adaptive sliding mode controller addresses *matched* and *unmatched* uncertainties on the arm plus object. Since the manipulator's motion may exert a torque on the object and make the object slip, we introduce a method for computing the maximum torques for the active actuators. This novel approach accounts for the static friction and the estimation error of the model, and it is computed using a minimisation problem. We additionally propose an inequality that represents a sufficient condition for tuning the controller's gains, which always attains no slip. A set of experiments are carried out using a (2 + 1) robot: 2 actuated joints plus the 1 passive joint (the connection gripper/object). Firstly, we show the performance of the proposed I-ASMC in the *point-tracking* and *trajectory-tracking* problem under the presence of *matched* and *unmatched* uncertainties. Secondly, we run the grasping selection algorithm on 150 grasping candidates with different force profiles and analysed the results. We also test the pipeline (grasp selection + controller) on a real scenario with a 7 dof robot constrained to planar movements. Although we study this manipulation problem in the particularly challenging planar case (the robot is non-STLC when the object starts moving) , our approach is straightforward to extend to any arm (with n joints) equipped with a clamping gripper and tasked with the grasp of an object under uncertainty.

CHAPTER 8

Conclusion

This work is motivated by the challenges of decommissioning and dismantling legacy nuclear facilities. The research has been funded by the UK Nuclear Decommissioning Authority with assistance from the UK National Nuclear Laboratory. The UK alone contains an estimated 4.9 million tonnes of nuclear waste, some of which dates back three-quarters of a century, including the Cold War era, and is associated with significant uncertainties. The NDA estimates that the UK legacy nuclear cleanup task will take more than 120 years, with an estimated cost of £130 billion.

Numerous ageing legacy nuclear facilities remain, many of which are severely contaminated with radioactive material. With time these structures are deteriorating, increasing the urgency to find ways to achieve decontamination, dismantlement, and transfer of contaminated materials into safe repositories. These facilities often contain complex pipework, vessels, glove-box chambers, and plant that must all be dismantled and cut into pieces for placing into waste storage containers. The current decommissioning plan calls for at least one million entries of human workers into hazardous zones, wearing air-fed plastic suits, to carry out complex and physically very demanding demolition work, *e.g.*

using heavy cutting and grinding tools. In other situations, radiation levels are far too high for humans, so remote machinery is the only option.

Robotic technologies that can speed up this process, and decrease the 120-year timeline, will: i) save large amounts of money for the tax-payer; ii) decrease the exposure of society to intolerable risks, associated with the continued existence of highly contaminated and deteriorating industrial structures and facilities; iii) remove human workers from hazardous environments; iv) eliminate very large quantities of secondary waste in the form of contaminated protective suits and Personal Protective Equipment (PPE), which must be used by human workers in hazardous zones; v) resolve challenges in zones which are too radioactive for any by humans at all, even using PPE.

This thesis addressed the planning and controlling of robots in extreme scenarios. The conditions in these environments pose a severe challenge to the state-of-the-art approach, and the best solution to deploy is yet an open question.

The task of decommissioning barrels containing nuclear wastes is common in nuclear facilities. Every container needs to be size-reduced using heavy machinery, *i.e.* rotatory tools, laser cutters, or hydraulic shears. Then, the contents are segregated according to radiation intensity, material, and size; or further reduced to achieve efficient storage.

Generally, these tasks have three phases planning, control, and grasping. The robot must generate a roadmap of configurations that allows the robot's end-effector to stay in touch with the surface, control the force/motion in either joint or operational space, and select the best method to grasp tools and/or objects to achieve a successful task.

Because of the criticality of working in these environments, the planning stage diverges from the canonical approaches. Whereas general operations demand the planner to attain the shortest path connecting two points, nuclear sites measure a successful operation differently. They positively accept a longer trajectory (*rough* path) if it allows a greater likelihood of achieving a complete execution.

Thereby, we propose a path planner called RRT*-RMM, which generates *rough* cutting paths for a fixed-based arm. The unstructured nature of the environment does not provide models of the objects to cut, and we perceive them via 3D cameras. The captured data are a noisy representation of the object's surface.

To reject the noisiness, we assume the object is smooth. This consideration allows using the Riemannian geometry to generate paths on top of the real object's surface.

Besides the vision issue, radioactive cells introduce several uncertainties in the robot's behaviour and the object's physical characteristics. Objects exposed to radiation have shown to change their dynamic properties. Therefore, we propose a novel metric to account for the manipulability of the robot throughout the trajectory. The improved manipulability guarantees that the robot has a margin while operating, and it depends only on kinematic coefficients.

Our metric is robot and planner agnostic, but we integrate it within the RRT* framework for testing the efficacy. An extensive set of experiments show that our approach yields maximum manipulability on a wide range of surfaces. Moreover, the robot inherently avoids singular configurations.

We report experiments on several point clouds recorded using a low-budget 3D camera. Moreover, we test the feasibility of the paths on a 7 dof Panda robot.

Although a few planners study the generation of paths with end-effector constraints, they fall short in accounting for the path's robustness and do not consider the manipulability throughout the planning. Moreover, they often resort to a model of the object and fail to generalise to noisy PCs.

In chapter 3, we extend our results to the class of non-holonomic mobile manipulators. Fixed-base robots have the shortcoming of a limited workspace, and they might fail to perform tasks far from the base. On the other hand, floating-base robots cover a virtually infinite range. However, these platforms cannot move instantaneously in any direction and have to comply with the mechanics. Therefore, we propose a projection method to include the non-holonomic property within our planner. We call the new algorithm RRT*-CRMM.

In the literature, researchers have investigated the control of NMM, but only a few works study the planning for NMM under task constraints. Our novel approach allows generating trajectories for non-holonomic mobile manipulators, where the end-effector must touch the object's surface while keeping the desired orientation and maximising the arm's manipulability.

We implement a multi-objective cost function that accounts for manipulability, travel distance, and base velocity. We analyse the effect of weighting the costs to obtain the best trade-off based on the application. Moreover, we run simulation experiments using a husky + UR5 robot. The algorithm yields great results on several objects and shows the capability to extend to non-smooth surfaces. We also report a performance comparison between RRT*-RMM and RRT*-CRMM, which shows the beneficial effect of using NMM.

In chapters 3 and 4, we reduce the manipulability to a scalar index. Although this is a convenient method for iterative purposes, the original manipulability is in matrix form. The manipulability matrix defines the residual capability to move or apply a force in each direction in the operational space.

A few researchers have begun to investigate how to leverage manipulability matrices for performing several tasks. However, they often limit the research to free motions, where the robot does not interact with the environment. Moreover, they rely on robot demonstrations to obtain profiles of manipulability matrices.

We propose to account for the shape of the manipulability matrix while planning robot-environment interaction tasks. Our method resorts to an objective functional that accounts for the manipulability shape and exploits a metric tailored to compute the distance between two matrices. We show that we can define the target matrix using robotic-naive user experience, task-information, or robot data.

Additionally, we account for the smoothness of the path throughout the trajectory and propose a hybrid force/motion controller that allows exploiting the improved manipulability showing compliant behaviours.

Thanks to the compliancy, the robot may move across areas that display higher frictions than estimated beforehand. Although hybrid force/motion controllers are present in the literature, they do not support improved manipulability. Our controller casts force and motion into separated spaces and exhibits compliant behaviours only in the motion component.

Alongside the planning problem, the robot must autonomously decide how to grasp an object for achieving a specific task. Many algorithms are present in the literature for generating grasp candidates onto the object's surface. However, they pivot around the generation of stable grasps and overlook the meaning behind the grasping. A body of work has proposed to select a grasp candidate accounting for metrics beyond stability. Although obtaining a force closure grasp is paramount, the task following the grasp stage is equally important.

In chapter 6, we propose an algorithm that selects the grasp candidate yielding a collision-free motion for the robot after the grasping. Other work follows this line and explores the post-grasping movements but focuses on different aspects *e.g.* safety, manipulability, etc.

In many situations, the task dictates the object trajectory, *e.g.* the screwdriver rotates on itself to drive a screw. Our algorithm exploits this knowledge and returns an index describing the collision cost throughout the task for every grasp candidate on the object. We consider a set of body points on every joint and link of the robot, and we compute the sum of distances between these body points and obstacles.

We performed a collection of experiments in simulation and in the real world using a Panda robot. Our algorithm successfully picks the grasp candidate more suitable to achieve the task at hand. We run tests on the time efficiency of our approach. It displays great real-time performance with a frequency of almost 60 Hz.

Besides our method, other researchers have proposed to select grasp candidates based on task metrics. We analysed four indexes and set up a minimisation problem that allows the prioritisation of a metric using weights. Although we would like to have the highest manipulability in the post-grasp trajectory, collision avoidance is more important. We base our experiments on a real PC and a set of grasp candidates generated by the GPD algorithm. Results show the effect of weights over the choice. It appears evident that the designing phase is critical to selecting the best set of coefficients based on the application.

Finally, in chapter 7, we addressed the problem of dual-arm handling. Although we have a trajectory for one arm, *e.g.* performing the cutting, the object must stay still throughout the execution. However, the grasp and control of a barrel under uncertainties are challenging.

Barrel exposed to radiation may have changed their physical characteristics. Therefore, we cannot assume to know the friction and softness of the object beforehand. The principal shortcoming of this effect is that we have no guarantee of obtaining a secure grasp after the gripper closure. Even more so when the robot is in particular configurations where the gravity term is missing.

Chapter 7 studied this challenging problem and proposed to tackle the problem from two sides. At first, we introduced a grasp selection algorithm, which picks the grasp candidate yielding lesser effort for the robot while stabilising it. Then, we propose an Integral Adaptive SMC that robustly controls the arm to the equilibrium. To assure the controller's gains are suitable for the friction limitations, we introduced a procedure that leads to selecting the fittest coefficients.

We mathematically proved the stability of the controller, and both simulation and real-world experiments confirmed our approach. We implement both the grasp selection strategy and IASMC on the Panda robot. However, we constrained the robot to only planar motions. This configuration allows studying the case of zero gravity on the interaction gripper/object, which is non-Small Time Local Controllable. A human operator must carry out a forceful task on the picked object, *e.g.* similarly to what a tooling arm would do, then the robot must account for the task to come, select the best grasp location, and stabilise the object throughout the task.

Experiments clearly show how the operations affect grasp selection. Moreover, we report the effect of picking a perfectly stable grasp that is not suitable for the following task.

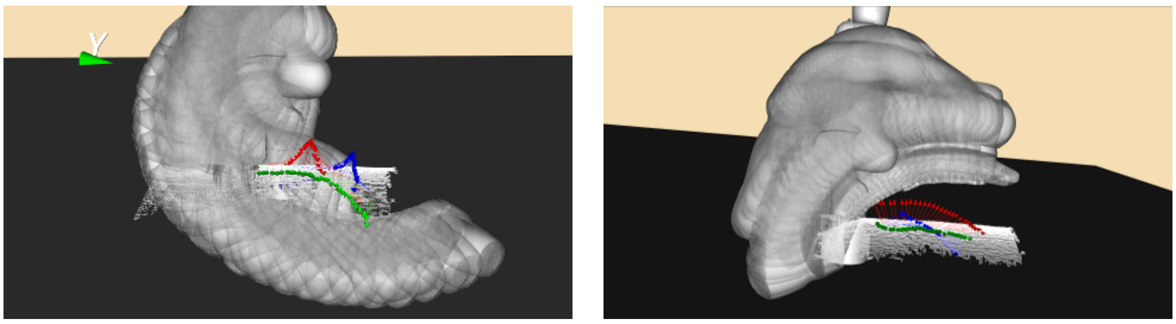

(a) (b)

Figure 8.1: The picture reports two failures of the RRT*-RMM algorithm due to normal estimation. The PCL's algorithm estimates correctly the right side of the PC, and on the opposite direction (upward rather than downward) on the left side.

8.1 Discussion and limitation

Throughout this thesis, we always assumed to interact with smooth surfaces, except for the folded cardboard experiment in Sec. 4.4. However, many objects have sharp edges and corners that undermine the smoothness assumption. This limitation bounds our approach to work on a subcategory of items and does not provide a straightforward extension to more general surfaces. A possible workaround for this issue is splitting the original surface into smaller areas, where the smoothness assumption holds. Then, we run the algorithm as many times as the number of newly created areas with the constraint on the initial and destination points.

On the other hand, this approach has the pitfall of generating suboptimal paths. Because we need to input the initial and destination points at every run, we must set a priori those points heuristically, which might negatively impact the overall optimisation.

A possible avenue to speed up the algorithm is to set all initial and destination points upfront and then run all algorithms in a parallel fashion. Oppositely, we might trade performance to improve the manipulability of the path. Although the initial end-effector's position must be fixed, the destination may be a region. Therefore, by running the planners one by one and setting the new initial position as the last destination point, we improve the obtained manipulability.

Future studies will dwell on how to release this restriction and extend the algorithms to non-Riemannian surfaces.

Furthermore, the smoothness assumption plays a secondary role. Our algorithms computed the normals to the surface using PCA. However, this method is prone to inaccuracies in vector estimation. PCA takes a set of points from the PC and fits a plane on the data. Fig. 8.1 reports two examples where our algorithm fails to obtain a feasible path due to a wrong estimation of the normal vector. The method from the PCL library estimates the normals upward on the right side and downward on the left side. Although the algorithm finds a solution and interpolates between two different orientations, the path is not feasible. We solved this problem by engineering a check at the end of the planning, which controls whether a collision between object and robot is present or not.

We may mitigate this problem by increasing the number of points considered by the method, which will yield a more robust behaviour against uncertainties. However, we encounter two issues: the method becomes time-consuming; and the estimated normal might yet be entirely off, *e.g.* computing the normal of an edge, increasing the points means to filter out the real corner.

When we perform offline operations, we can speed up the process by computing all the normals beforehand and storing them in a structure. The algorithm will address the data only when needed in a hash-table fashion.

Besides the help of a few loopholes, the problem is still an open challenge, and future work will analyse the impact of the inaccuracies and explore new methods to improve the computation of the surface's normals.

RRT*-RMM and RRT*-CRMM used the differential kinematics to generate new configurations for the robot. At each step, we computed a new e-e pose (and consequently the configuration) by small increments from the last pose. Thereby, the initial configuration of the robot influences the entire performance of the algorithm. If the robot is redundant, more than one configuration exists for a given position and orientation. In future works, we will further study how to obtain initial configurations that reduce this effect.

Application-wise, we believe that industrial operations that demand continuous e-e/surface contact are still understudied, *e.g.* welding, where the torch must touch the object to provide long-lasting welds. And we plan to extend our methods by integrating with grasping approaches to perform mobile manipulation tasks, like the planning of arbitrary heavy-lifting.

In Sec. 5, we assumed that the target manipulability ellipsoid is provided by technicians, who give insight on the task to perform. However, we will investigate more sophisticated methodologies in the future. For instance, a study on human behaviours may shed light on the most common manipulability ellipsoids. The skills learnt by human demonstration may result in a collection of templates that we may query for addressing multiple tasks.

This approach would allow naive users to generalise the algorithm for any force-full task without core knowledge of the optimisation.

Among the unlimited opportunities yielded by the grasp selection, we will study the definition of a unified framework for re-ranking grasp candidates. It would allow the simple creation of a hierarchy between metrics based on the task at hand. Furthermore, we believe a fair benchmarking procedure is currently missing in the literature. We will then promote the study of a collection of tests to assess a grasp generator based on both canonical methods and post-grasp motions.

Although this thesis does not perform any actual cutting operations, we set the ground for deploying robotic solutions in these hazardous settings. Future work will investigate specific cutting tools and materials properties to roll out full-force tasks. Moreover, we will study how to estimate the friction of an object online to avoid slippage. It would ease the constraint on the gain selection for the IASMC and allow for obtaining less conservative results.

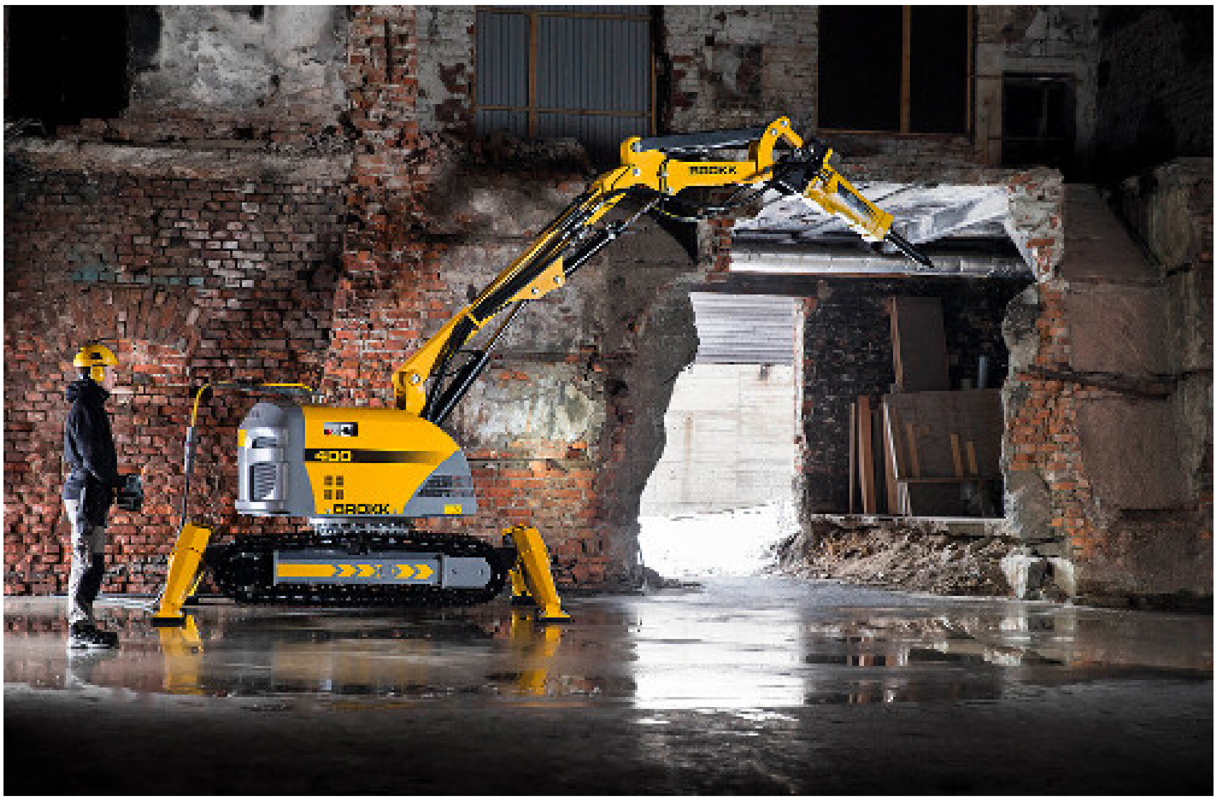


Figure 8.2: The Brook company is one of the leader in the field of robotic demolition. The picture shows a robot teleoperated by the human operator for dismantling an old building.

8.2 Applications of this thesis

The tackling of nuclear applications fueled the research effort of this thesis. Human operators are the principal source of labour for tasks in such dangerous scenarios, and the risk for their health in both the short- and long-term range may have catastrophic consequences. However, nuclear environments are not the only recipients of the methods developed in this thesis.

Our approaches are suitable for many applications where the robot must interact with the environment and inconsistent perception data is present. Robotics solutions are inching into several domains spreading from construction to underwater exploration.

The demolition of a house requires operators to dismantle the building using jackhammers and flatten out every standing piece. Although part of the job is achieved using remote-operated wrecking balls that quickly remove most of the construction, a more accurate swipe is necessary to demolish the remaining parts *i.e.* pillars and walls.

Robots like the picture 8.2, manufactured by Brokk Global, provides operators with a new method to carry out demolition tasks. Those robots exert forces far greater than what a human may sustain using a jackhammer, and their crawlers allow an easy traversing of several terrains.

The proposed path planners are a perfect fit for these applications. The robot must interact with an external surface with no prior knowledge, *e.g.* walls, pillars, tunnels, etc. Because the goal of the task is demolition, a departure from the shortest path is accepted. Moreover, the tool provides a clear direction of use which we can exploit to infer the best manipulability ellipsoid throughout the task.

Another instance of an application that may benefit from our work is the hull-cleaning. Periodically, every submerged (*e.g.* offshore station) or partially submerged (*e.g.* boat) objects must be cleaned from marine elements like algae, marine life, and waterline scum.

As for vessels, if the boat is small and docked, the sailor would use bespoke instruments to scrape the residue from the hull. However, if the ship is big or floating at sea, they need to dive into the water. Thereby, they hire divers that will swipe the full range of the hull using cleaning devices.

Recently, many companies, like the ECA Group, have started to develop tethered Remotely Operated underwater Vehicles (ROV) for hull-cleaning. The ROVs mount brushes that scrape out the dirt from the vessel's surface. Although those surfaces are smooth, the underwater perception creates many outliers and noise in the readings.

The algorithms proposed in this thesis rely on the Riemannian geometry to deal with uncertainties at the perception level, which yields a deployable solution to the hull-cleaning problem. Thanks to our methods, we may generate paths for the ROV that constrain the robot to stay on the arbitrary surface and reject the disturbances.

In the medical domain, robots like the Da Vinci allow surgeons to perform complicated procedures with the least invasive requirements. For safety reasons, these robots move only in teleoperation. However, the logic behind the arms' movements may greatly benefit from our approaches.

In stitching operations, the surgeon must grasp the semi-circular needle and pierce the human's flesh to pass the suture thread. Generally, the surgeon would grasp the needle to accommodate the inserting operation and maximise its dexterity.

Once the surgeon begins to teleoperate robotic arms, training is necessary to teach how to perform the stitch using the new machine. In teleoperation, the operator is bound to the robot's kinematics and must adapt the design of grasping the needle. Haptic feedback proved to provide valuable information to the operator, but the elaboration of multiple cues has also been shown to produce a high cognitive load on the user.

Our techniques address two issues: we may propose grasp candidates on the needle for stitching operations and provide the surgeon with multiple incision paths.

Because we know the needle's trajectory, the algorithm in Chp. 6 may propose a set of grasp candidates on the needle, which would result in a collision-free path for the robot. This approach would lift a share of the cognitive burden from the surgeon, who can centre herself on the patient.

Contrarily to other domains, long departures from the shortest path are not positively accepted in surgery. Concurrently, it is almost impossible to obtain a clear straight line because of the skin imperfection, and few modifications are allowed.

Thanks to the coefficients, we can dial our planners' weights to generate paths having only a slim divergence from the shortest path. The proposed trajectories would still have improved manipulability, and the surgeon may pick the one more suitable for the incision at hand.

8.3 Other work

Although my principal line of research is about the handling of objects in hazardous scenarios, I have pursued several collaborations with fellow researchers from other universities and extended my work to other domains.

To bench-marking [BMR$^+$20]: although many grasp generators exist in the literature, the robotics community lacks a consensus on how to compare two grasp algorithms. In industry, engineers often resort to the Max Picks Per Hour (MPPH) to assess the performance of the grasp generator. They count the successful grasps for a specific pair object/gripper over an hour time. Although this approach has the benefit of being easy to measure, it is not suitable for all grippers and tasks. Moreover, the trade-off between being fast and efficient plays a critical role. On the contrary, academia proposes wider metrics, which fit in more grippers. However, they have the pitfall of being tailored to either the gripper or the task at hand. Only a small body of work compares the proposed grasp algorithm with others in a fair fashion, and no shared benchmarking methodology is available. Therefore, we propose a method for assessing a grasping algorithm using a set of tasks that are robot and gripper agnostic. The tasks account for both dynamic movements (shaking an object after grasp) and static (lifting and placing the object). Moreover, our method permits obtaining a measure of the performance with a variety of objects. We considered the YCB dataset and picked objects with several degrees of complexity, size, and material. Also, we propose to assess the grasping algorithm on deformable objects.

To teleoperation [PKM$^+$20]: The work in Sec. 6.2 allows the robot to select the grasp candidates that yields a collision-free motion for performing the task after the grasp. We leveraged the information provided by this algorithm to ease the cognitive load of a human operator in a teleoperation scenario. In many applications, full robotic automation is yet to be reached, and human workers must actively control the robot to perform a task. One of the most common control paradigms is teleoperation, where the worker manoeuvres a haptic device to move the end-effector of the robot and pick up objects. On the other hand, this approach poses two limitations: it requires knowledge of the robot kinematic to avoid running into singular configurations; it poses a high cognitive load on the operator. We then propose a strategy that provides cues to the operator and drives the user to the grasp candidate that best matches the collision-avoidance metric. A feedback force is produced by the haptic device and aids the worker to select the best grasp. Thanks to this strategy, the operator has the twofold benefit of staying always in control of the robot and possibly rejecting the suggested grasp candidate while decreasing the burden to choose the most suitable grasp for the task ahead.

To robot-human handover [OCP$^+$20]: Industry 4.0 is creating a momentum for using collaborative robots (co-bots). More and more companies are deploying robotic companions alongside human labours to improve the efficiency of the production line. The robot partner must pass over objects to the operator, *i.e.* a screwdriver, a drill, etc. and be able to receive objects from the human. Another instance where robotic partners are becoming critical is for care-taking. Service robot needs to help elders by handing over a glass of water to let them swallow their medicines, or by moving kitchen objects *i.e.* ladles and bowls. Position and orientation of the point of grasp, the realise strategy, or the safety issues for avoiding collision with the operator are only the tip of the complex problem posed by the task of handing over an object. A particularly interesting element is how the passing over of an object affects the task efficiency. Generally, we would like to grasp the object in such a way that we can use it without further manipulation. In order to reach this degree of efficiency, the robot must know beforehand the task to perform with the object and select a suitable grasp. This will allow the receiver to grasp the object from the "graspable part" and use the "functional part" to perform the task, *e.g.* the screwdriver's handle is the "graspable part", whereas the tip of the screwdriver is the "functional part". We, therefore, explored this concept with a study involving 23 human subjects, where a robot arm would hand over tools to the human. The human was tasked with grasping the object and performing a task afterwards. Two sets of data were collected: qualitative, every participant was asked to answer a questionnaire after every task; quantitative, an IMU sensor was attached to the participant's wrist. The data have shown that a smart grasp selection is perceived better by the user, who looks at the robot as collaborative. Moreover, the quantitative data have shown a smaller time for the user to decide how to grasp the object, which results in less time to perform the task.

To grasping [AOP+21]: Only a few grasp generators in the literature are ready to use. Despite the researchers' effort in producing new methods to synthesise grasp candidates on several objects, these approaches often fail to escape research labs. However, software capable of providing ready-to-use grasp candidates would greatly speed up both the research and industrial processes.

We proposed a grasp generator based on the Unity physics engine. The robustness of this platform yields great accuracy in the results and narrows down the gap between real scenarios and simulations. Non-homogeneous frictions, dynamics, slippage, and collision detection are only a few pivotal features provided by Unity.

PrendoSim deploys a technique called proxy-hand for testing candidates. We consider two twin grippers, one kinematic and one kinetic. Firstly, we place the object before the open gripper with a random orientation. Secondly, we commence the closure phase. The two grippers behave differently. The kinematic gripper stops once one digit touches the object. Instead, the kinetic gripper keeps closing even after the first contact. However, an elastic force proportional to the displacement between the two grippers is applied. The closure phase ends when both grippers are at rest. To check if the grasp candidate is reliable, we activate the object's dynamics and test if the grasp holds. If the test passes, we store the grasp candidate, and we move to the next one.

All these procedures are transparent to the user. Our software builds a friendly user interface on top of this framework and guides the user through the loading, selection, and storing of the data straightforwardly. Moreover, all data are conveniently stored in an XML file. Thanks to this structure, our approach allows generating quickly several grasp candidates on an object, which are either readily deployable for real applications or stored for creating datasets. The released version of PrendoSim makes available three grippers and a selection of objects. On the other hand, users may define bespoke objects with several materials and frictions.

APPENDIX A

Projects repositories

- Implementation of the KD-Tree algorithm with multi objective criteria `https://gitlab.com/t.pardi/KDtree_MultiObj`
- *RRT** path planner implementation accounting for the manipulability of the robot `https://gitlab.com/t.pardi/RRTStar-RMM`
- RKDLib is a wrapper of the KDL library from Orocos, which uses Eigen and simplify the obtaining of kinematics and dynamics properties for any robot by URDF. `https://gitlab.com/t.pardi/rkdlib`
- Software for using custom controllers using the API of the Panda Emika robot. `https://gitlab.com/t.pardi/panda_control`

Bibliography

[AD14] E. Aertbeliën and J. De Schutter. etasl/etc: A constraint-based task specification language and robot controller using expression graphs. In *2014*

IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 1540–1546, 2014.

[AJM$^+$18] Abhinav Agarwalla, Arnav Kumar Jain, K V Manohar, Arpit Tarang Saxena, and Jayanta Mukhopadhyay. Bayesian optimisation with prior reuse for motion planning in robot soccer. In *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*, CoDS-COMAD '18, page 88–97, New York, NY, USA, 2018. Association for Computing Machinery.

[AMK$^+$16] Amir M. Ghalamzan E., Nikos Mavrakis, Marek Kopicki, Rustam Stolkin, and Ales Leonardis. Task-relevant grasp selection: A joint solution to planning grasps and manipulative motion trajectories. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 907–914, 2016.

[AMO$^+$18] Maxime Adjigble, Naresh Marturi, Valerio Ortenzi, Vijaykumar Rajasekaran, Peter Corke, and Rustam Stolkin. Model-free and learning-free grasping by local contact moment matching. In *IEEE-RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2018.

[AOP$^+$21] D. Abdlkarim, V. Ortenzi, T. Pardi, M. Filipovica, A.M. Wing, and M. Di Luca K. J. Kuchenbecker. PrendoSim: Proxy-Hand-Based Robot Grasp Generator, 2021.

[Ass20] World Nuclear Association. Decommissioning Nuclear Facilities. `https://www.world-nuclear.org/information-library/nuclear-fuel-cycle/nuclear-wastes/decommissioning-nuclear-facilities.aspx`, 2020.

[AtPP17] K. Saenko A. ten Pas, M. Gualtieri and R. Platt. Grasp pose detection in point clouds. *The International Journal of Robotics Research*, 36(13-14):1455–1473, 2017.

[Aut19] Nuclear Decommissioning Authority. Nuclear Provision: the cost of cleaning up Britain's historic nuclear sites. `https://www.gov.uk/government/publications/` `nuclear-provision-explaining-the-cost-of-cleaning-up-britains-nuclear` `nuclear-provision-explaining-the-cost-of-cleaning-up-britains-nuclear` 2019.

[AW96] Nancy M Amato and Yan Wu. A randomized roadmap method for path and manipulation planning. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 113–120. IEEE, 1996.

[BA15] Patrick Beeson and Barrett Ames. Trac-ik: An open-source library for improved solving of generic inverse kinematics. In *IEEE-RAS International Conference on Humanoid Robots*, pages 928–935. IEEE, 2015.

[BDN$^+$07] D. Berenson, R. Diankov, Koichi Nishiwaki, Satoshi Kagami, and J. Kuffner. Grasp planning in complex scenes. In *IEEE-RAS International Conference on Humanoid Robots*, pages 42–48, Nov 2007.

[Bha07] Rajendra Bhatia. *Positive Definite Matrices*. Princeton University Press, 2007.

[BKDA06] Dominik Bertram, James Kuffner, Rüdiger Dillmann, and Tamim Asfour. An integrated approach to inverse kinematics and path planning for redundant manipulators. In *IEEE International Conference in Robotic and Automation*, pages 1874–1879, 2006.

[BMK20] L. Berscheid, P. Meißner, and T. Kröger. Self-supervised learning for precise pick-and-place without object model. *IEEE Robotics and Automation Letters*, 5(3):4828–4835, 2020.

[BMR$^+$20] Y. Bekiroglu, N. Marturi, M. A. Roa, K. J. M. Adjigble, T. Pardi, C. Grimm, R. Balasubramanian, K. Hang, and R. Stolkin. Benchmarking protocol for grasp planning algorithms. *IEEE Robotics and Automation Letters*, 5(2):315–322, 2020.

[BSK11] Dmitry Berenson, Siddhartha Srinivasa, and James Kuffner. Task space regions: A framework for pose-constrained manipulation planning. *The International Journal of Robotics Research*, 30(12), 2011.

[CFD18] L. Chen, L. F. C. Figueredo, and M. Dogar. Manipulation planning under changing external forces. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3503–3510, 2018.

[CGF$^+$14] M.G. Catalano, G. Grioli, E. Farnioli, A. Serio, C. Piazza, and A. Bicchi. Adaptive synergies for the design and control of the pisa/iit softhand. *The International Journal of Robotics Research*, 33(5):768–782, 2014.

[CJS07] J. Cortes, L. Jaillet, and T. Simeon. Molecular disassembly with rrt-like algorithms. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3301–3306, 2007.

[CK89] Ricardo Carelli and Rafael Kelly. Adaptive control of constrained robots modeled by singular systems. In *Proceedings of the IEEE Conference on Decision and Control*, 1989.

[COCC19] F. Cini, V. Ortenzi, P. Corke, and M. Controzzi. On the choice of grasp type and location when handing over an object. *Science Robotics*, 4(27), 2019.

[CS17] S. Cruciani and C. Smith. In-hand manipulation using three-stages open loop pivoting. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1244–1251, 2017.

[DB08] R. Daily and D. M. Bevly. Harmonic potential field path planning for high speed vehicles. In *2008 American Control Conference*, pages 4609–4614, 2008.

[DH06] A. M. Dollar and R. D. Howe. A robust compliant grasper via shape deposition manufacturing. *IEEE/ASME Transactions on Mechatronics*, 11(2):154–161, 2006.

[DHW$^+$17] Feng Ding, Jian Huang, Yongji Wang, Junmin Zhang, and Shunfan He. Sliding mode control with an extended disturbance observer for a class of underactuated system in cascaded form. *Nonlinear Dynamics*, 2017.

[DLW01] Dan Ding, Yun-Hui Lee, and Shuguo Wang. Computation of 3-d form-closure grasps. *IEEE Transactions on Robotics and Automation*, 17(4):515–522, 2001.

[DMO00] Alessandro De Luca, Raffaella Mattone, and Giuseppe Oriolo. Stabilization of an underactuated planar 2R manipulator. *International Journal of Robust and Nonlinear Control*, 2000.

[DOG07] A. De Luca, G. Oriolo, and P. R. Giordano. Image-based visual servoing schemes for nonholonomic mobile manipulators. *Robotica*, 25(2):131–145, 2007.

[DPM17] Renaud Detry, Jeremie Papon, and Larry Matthies. Task oriented grasping with semantic and geometric scene understanding. In *IEEE-RSJ International Conference on Intelligent Robots and Systems*, pages 3266–3273. IEEE, 2017.

[DR20] Jonathan M Dodds and John Rawcliffe. Radionuclide distribution during ytterbium doped fibre laser cutting for nuclear decommissioning. *Progress in Nuclear Energy*, 118:103122, 2020.

[DSMB95] Keith L. Doty, Eric M. Schwartz, Claudio Melchiorri, and Claudio Bonivento. Robot Manipulability. *IEEE Transactions on Robotics and Automation*, 1995.

[FC92] Carlo Ferrari and John Canny. Planning optimal grasps. In *IEEE-RSJ International Conference on Robotics and Automation*, pages 2290–2295. IEEE, 1992.

[GA20] Alejandro Gutiérrez–Giles and Marco Arteaga–Pérez. Output feedback hybrid force/motion control for robotic manipulators interacting with unknown rigid surfaces. *Robotica*, 38(1):136–158, 2020.

[GEAFRGS17] Amir M. Ghalamzan E., Firas Abi-Faraj, Paolo Robuffo Giordano, and Rustam Stolkin. Human-in-the-loop optimisation: mixed initiative grasping for optimally facilitating post-grasp manipulative actions. In *IEEE-RSJ International Conference on Intelligent Robots and Systems*, page submitted, 2017.

[GFC18] Emile Glorieux, Pasquale Franciosa, and Darek Ceglarek. End-effector design optimisation and multi-robot motion planning for handling compliant parts. *Structural and Multidisciplinary Optimization*, 2018.

[GMS17] A.M.E. Ghalamzan, N. Mavrakis, and R. Stolkin. Grasp that optimises objectives along post-grasp trajectories. In *2017 5th RSI International Conference on Robotics and Mechatronics (ICRoM)*, pages 51–56, 2017.

[GR17] Marcus Greiff and Anders Robertsson. Optimisation-based motion planning with obstacles and priorities. *IFAC-PapersOnLine*, 50(1):11670 – 11676, 2017. 20th IFAC World Congress.

[GSB14] Jonathan D. Gammell, Siddhartha S. Srinivasa, and Timothy D. Barfoot. Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. In *Intelligent Robots and Systems*, pages 2997–3004. IEEE, 2014.

[GtPSP16] M. Gualtieri, A. ten Pas, K. Saenko, and R. Platt. High precision grasp pose detection in dense clutter. In *IEEE-RSJ International Conference on Intelligent Robots and Systems*, pages 598–605, 2016.

[GZA18] J. Gao, Y. Zhou, and T. Asfour. Projected force-admittance control for compliant bimanual tasks. In *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, pages 1–9, 2018.

[HA92] Y. K. Hwang and N. Ahuja. A potential field approach to path planning. *IEEE Transactions on Robotics and Automation*, 8(1):23–32, Feb 1992.

[HA01] L Han and NM Amato. A kinematics-based probabilistic roadmap method for closed chain systems: Li han, texas a nancy m. amato, texas a. In *Algorithmic and Computational Robotics*, pages 243–251. AK Peters/CRC Press, 2001.

[HFY20] S. D. Han, S. W. Feng, and J. Yu. Toward fast and optimal robotic pick-and-place on a moving conveyor. *IEEE Robotics and Automation Letters*, 5(2):446–453, 2020.

[HJJM16] Yifan Hou, Zhenzhong Jia, A. M. Johnson, and M. T. Mason. Robust planar dynamic pivoting by regulating inertial and grip forces. In *WAFR*, 2016.

[Hog84] N. Hogan. Impedance control: An approach to manipulation. In *1984 American Control Conference*, pages 304–313, 1984.

[Hog87] N. Hogan. Stable execution of contact tasks using impedance control. In *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, volume 4, pages 1047–1054, 1987.

[Hop08] Hugues Hoppe. Poisson surface reconstruction and its applications. In *Proceedings of the 2008 ACM Symposium on Solid and Physical Modeling*, SPM '08, page 10, New York, NY, USA, 2008. Association for Computing Machinery.

[JCS08] Léonard Jaillet, Juan Cortés, and Thierry Siméon. Transition-based RRT for path planning in continuous cost spaces. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2145–2150, 2008.

[JCS10] L. Jaillet, J. Cortés, and T. Siméon. Sampling-based path planning on configuration-space costmaps. *IEEE Transactions on Robotics*, 26(4):635–646, Aug 2010.

[JH19] E. Jelavic and M. Hutter. Whole-body motion planning for walking excavators. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2292–2299, 2019.

[JHK19] M. Jorda, E. G. Herrero, and O. Khatib. Contact-driven posture behavior for safe and interactive robot operation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9243–9249, 2019.

[JLYS11] T. Ju, S. Liu, J. Yang, and D. Sun. Apply rrt-based path planning to robotic manipulation of biological cells with optical tweezer. In *2011 IEEE International Conference on Mechatronics and Automation*, pages 221–226, 2011.

[JRCC0] Noémie Jaquier, Leonel Rozo, Darwin G Caldwell, and Sylvain Calinon. Geometry-aware manipulability learning, tracking, and transfer. *The International Journal of Robotics Research*, 0(0):0278364920946815, 0.

[KCT$^+$11] Mrinal Kalakrishnan, Sachin Chitta, Evangelos Theodorou, Peter Pastor, and Stefan Schaal. Stomp: Stochastic trajectory optimization for motion planning. In *2011 IEEE international conference on robotics and automation*, pages 4569–4574. IEEE, 2011.

[KDA$^+$16a] M. Kopicki, R. Detry, M. Adjigble, R. Stolkin, A. Leonardis, and J.L. Wyatt. One-shot learning and generation of dexterous grasps for novel objects. *International Journal of Robotics Research*, 35(8):959–976, 2016.

[KDA$^+$16b] Marek Kopicki, Renaud Detry, Maxime Adjigble, Rustam Stolkin, Ales Leonardis, and Jeremy L. Wyatt. One-shot learning and generation of dexterous grasps for novel objects. *The International Journal of Robotics Research*, 35(8):959–976, 2016.

[KF11] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894, 2011.

[KFT$^+$08] Y. Kuwata, G. A. Fiore, J. Teo, E. Frazzoli, and J. P. How. Motion planning for urban driving using rrt. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1681–1686, 2008.

[KG13] N. Kammerer and P. Garrec. Dry friction modeling in dynamic identification for robot manipulators: Theory and experiments. In *2013 IEEE International Conference on Mechatronics (ICM)*, pages 422–429, 2013.

[Kha86] Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. In *Autonomous robot vehicles*, pages 396–404. Springer, 1986.

[Kha88] Oussama Khatib. Object manipulation in a multi-effector robot system. In *Proceedings of the 4th international symposium on Robotics Research*, pages 137–144, 1988.

[KJL18] K.D. Kallu, W. Jie, and M.C.Y. Lee. Sensorless reaction force estimation of the end effector of a dual-arm robot manipulator using sliding mode control with a sliding perturbation observer. *Int. J. Control Autom. Syst.*, 16(7):1367–1378, 2018.

[KK88] R. K. Kankaanranta and H. N. Koivo. Dynamics and simulation of compliant motion of a manipulator. *IEEE Journal on Robotics and Automation*, 4(2):163–173, 1988.

[KL00] J. J. Kuffner and S. M. LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, volume 2, pages 995–1001 vol.2, 2000.

[KMCY04] Jinhyun Kim, Giacomo Marani, Wan Kyun Chung, and Junku Yuh. A general singularity avoidance framework for robot manipulators: Task reconstruction method. In *Proceedings - IEEE International Conference on Robotics and Automation*, 2004.

[KRJS17] Jinu Krishnan, U.P. Rajeev, J. Jayabalan, and D.S. Sheela. Optimal motion planning based on path length minimisation. *Robotic Automation Systems*, 94:245 – 263, 2017.

[KSB16] D. Kappler, S. Schaal, and J. Bohg. Optimizing for what matters: The top grasp hypothesis. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2167–2174, 2016.

[KSBB07] Dov Kruger, Rustam Stolkin, Aaron Blum, and Joseph Briganti. Optimal auv path planning for extended missions in complex, fast-flowing estuarine environments. In *IEEE International Conference on Robotics and Automation*, pages 4265–4270. IEEE, 2007.

[KSLO96] L. E. Kavraki, P. Svestka, J. . Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[KUSP16] Beobkyoon Kim, Terry Taewoong Um, Chansu Suh, and F. C. Park. Tangent bundle RRT: A randomized algorithm for constrained motion planning. *Robotica*, 2016.

[KWP$^+$11] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller. Anytime motion planning using the rrt*. In *2011 IEEE International Conference on Robotics and Automation*, pages 1478–1483, 2011.

[LaV06] Steven M. LaValle. *Planning Algorithms.* Cambridge University Press, New York, NY, USA, 2006.

[LB18] Yu-Chi Lin and Dmitry Berenson. Humanoid navigation planning in large unstructured environments using traversability - based segmentation. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 7375–7382, 2018.

[LCLX18] Y. Li, R. Cui, Z. Li, and D. Xu. Neural network approximation based near-optimal motion planning with kinodynamic constraints using rrt. *IEEE Transactions on Industrial Electronics*, 65(11):8718–8729, 2018.

[LH03] P. Leven and S. Hutchinson. Using manipulability to bias sampling during the construction of probabilistic roadmaps. *IEEE Transactions on Robotics and Automation*, 19(6):1020–1026, 2003.

[LHSG$^+$19] Carlos Lucas, Daniel Hernández-Sosa, David Greiner, Aleš Zamuda, and Rui Caldeira. An approach to multi-objective path planning optimization for underwater gliders. *Sensors*, 19(24):5506, 2019.

[LIMO03] Alessandro De Luca, Stefano Iannitti, Raffaella Mattone, and Giuseppe

Oriolo. Underactuated Manipulators: Control Properties and Techniques. *Control*, 2003.

[Liu17] Jinkun Liu. Chapter 1 - Basic sliding mode control principle and design. In Jinkun Liu, editor, *Sliding Mode Control Using MATLAB*, pages 1–29. Academic Press, 2017.

[LN14] Godinho Leonor and José Natário. An introduction to riemannian geometry. *Universitex, ISBN 9783319086668*, 2014.

[LSB$^+$18] H. Lin, J. Smith, K. K. Babarahmati, N. Dehio, and M. Mistry. A projected inverse dynamics approach for multi-arm cartesian impedance control. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5421–5428, 2018.

[MAS$^+$16] Nikos Mavrakis, Amir M. Ghalamzan E., Rustam Stolkin, Luca Baronti, Marek Kopicki, and Marco Castellani. Analysis of the inertia and dynamics of grasped objects, for choosing optimal grasps to enable torque-efficient post-grasp manipulations. In *IEEE-RAS International Conference on Humanoid Robots*, pages 171–178. IEEE, 2016.

[MGER17] Nikos Mavrakis, Amir M. Ghalamzan E., and Stolkin Rustam. Safe robotic grasping: Minimum impact-force grasp selection. In *IEEE-RSJ International Conference on Intelligent Robots and Systems*, pages 4034–4041. IEEE, 2017.

[MGES$^+$16] Nikos Mavrakis, Amir M. Ghalamzan E., Rustam Stolkin, Luca Baronti, Marek Kopicki, and Marco Castellani. Analysis of the Inertia and Dynamics of Grasped Objects , for Choosing Optimal Grasps to Enable Torque-Efficient Post-Grasp Manipulations. *IEEE-RAS International Conference on Humanoid Robots*, pages 171–178, 2016.

[MJM01] Min Gyu Park, Jae Hyun Jeon, and Min Cheol Lee. Obstacle avoidance for mobile robots using artificial potential field approach with simulated annealing. In *ISIE 2001. 2001 IEEE International Symposium on Industrial Electronics Proceedings (Cat. No.01TH8570)*, volume 3, pages 1530–1535 vol.3, 2001.

[ML04] Lörinc Márton and Béla Lantos. Sliding Mode Robot Control with Friction and Payload Estimation. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 8(5):553–561, 2004.

[MRR$^+$17] Naresh Marturi, Alireza Rastegarpanah, Vijaykumar Rajasekaran, Valerio Ortenzi, Yasemin Bekiroglu, Jeffrey Kuo, and Rustam Stolkin. Towards advanced robotic manipulation for nuclear decommissioning. In *Robots Operating in Hazardous Environments*. IntechOpen, 2017.

[MS07] N. A. Melchior and R. Simmons. Particle rrt for path planning with uncertainty. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 1617–1624, 2007.

[MT17] Anirudha Majumdar and Russ Tedrake. Funnel libraries for real-time robust feedback motion planning. *The International Journal of Robotics Research*, 36(8):947–982, 2017.

[MXJ19] Xiaoqian Mu, Yuechuan Xue, and Y. Jia. Robotic cutting: Mechanics and control of knife motion. *2019 International Conference on Robotics and Automation (ICRA)*, pages 3066–3072, 2019.

[NAM06] Mehdi Nikkhah, Hashem Ashrafiuon, and Kenneth R. Muske. Optimal sliding mode control for underactuated systems. In *Proceedings of the American Control Conference*, 2006.

[NHGT02] K. Nagatani, T. Hirayama, A. Gofuku, and Y. Tanaka. Motion planning for mobile manipulator with keeping manipulability. In *IEEE International Conference on Intelligent Robots and Systems*, 2002.

[NKH16] Iram Noreen, Amna Khan, and Zulfiqar Habib. Optimal path planning using rrt* based approaches: A survey and future directions. *International Journal of Advanced Computer Science and Applications*, 7(11), 2016.

[NVTK03] I. K. Nikolos, K. P. Valavanis, N. C. Tsourveloudis, and A. N. Kostaras. Evolutionary algorithm based offline/online path planner for uav navigation. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 33(6):898–912, Dec 2003.

[OCP$^+$20] Valerio Ortenzi, Francesca Cini, Tommaso Pardi, Naresh Marturi, Rustam Stolkin, Peter Corke, and Marco Controzzi. The grasp strategy of a robot passer influences performance and quality of the robot-human object handover. *Frontiers in Robotics and AI*, 7:138, 2020.

[OS00] R. Olfati-Saber. Cascade normal forms for underactuated mechanical systems. *Proceedings of the IEEE Conference on Decision and Control*, 3:2162–2167, 2000.

[Pag99] Prabhakar R. Pagilla. Robotic surface finishing processes: Modeling, control, and experiments. *Journal of Dynamic Systems, Measurement, and Control*, 123, 1999.

[Paj15] Grzegorz Pajak. End-effector vibrations reduction in trajectory tracking for mobile manipulator. *Journal of Vibroengineering*, 17(1):101–111, 2015.

[PKM$^+$20] S. Parsa, D. Kamale, S. Mghames, K. Nazari, T. Pardi, A. R. Srinivasan, G. Neumann, M. Hanheide, and G. E. Amir. Haptic-guided shared control grasping: collision-free manipulation. In *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, pages 1552–1557, 2020.

[POF$^+$20a] T. Pardi, V. Ortenzi, C. Fairbairn, T. Pipe, A. M. G. Esfahani, and R. Stolkin. Planning maximum-manipulability cutting paths. *IEEE Robotics and Automation Letters*, 5(2):1999–2006, April 2020.

[POF$^+$20b] T. Pardi, V. Ortenzi, C. Fairbairn, T. Pipe, A. M. G. Esfahani, and R. Stolkin. Planning maximum-manipulability cutting paths. *IEEE Robotics and Automation Letters*, 5(2):1999–2006, 2020.

[PP14a] Grzegorz Pajak and Iwona Pajak. Collision-free trajectory planning for mobile manipulators subject to control constraints. *Archive of Mechanical Engineering*, 61(1):35–55, 2014.

[PP14b] Grzegorz Pajak and Iwona Pajak. Motion planning for mobile surgery assistant. *Acta of bioengineering and biomechanics*, 16(2), 2014.

[PP15] Grzegorz Pajak and Iwona Pajak. Sub-optimal trajectory planning for mobile manipulators. *Robotica*, 33(6):1181–1200, 2015.

[PPL$^+$20] Tommaso Pardi, M. Poggiani, E. Luberto, Alessandro Raugi, M. Garabini, Riccardo Persichini, M. Catalano, G. Grioli, M. Bonilla, and A. Bicchi. A soft robotics approach to autonomous warehouse picking. In *Advances on Robotic Item Picking Applications in Warehousing & E-Commerce Fulfillment*, pages 23–35. Springer International Publishing, 2020.

[PPM12] C. Park, Jia Pan, and D. Manocha. Itomp: Incremental trajectory optimization for real-time replanning in dynamic environments. In *ICAPS*, 2012.

[PS12] Farzin Piltan and Nasri B. Sulaiman. Review of sliding mode control of robotic manipulator. *World Applied Sciences Journal*, 18(12):1855–1869, 2012.

[PYPY18a] Y. Pan, C. Yang, L. Pan, and H. Yu. Integral sliding mode control: Performance, modification, and improvement. *IEEE Transactions on Industrial Informatics*, 14(7):3087–3096, 2018.

[PYPY18b] Yongping Pan, Chenguang Yang, Lin Pan, and Haoyong Yu. Integral Sliding Mode Control: Performance, Modification, and Improvement. *IEEE Transactions on Industrial Informatics*, 14(7):3087–3096, 2018.

[Qia99] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 1999.

[RAFGP19] Rahaf Rahal, Firas Abi-Farraj, Robuffo Paolo Giordano, and Claudio Pacchierotti. Haptic shared-control methods for robotic cutting under nonholonomic constraints. *IROS*, pages 8151–8157, 2019.

[RJCC17] Leonel Rozo, Noemie Jaquier, Sylvain Calinon, and Darwin G. Caldwell. Learning manipulability ellipsoids for task compatibility in robot manipulation. *IEEE International Conference on Intelligent Robots and Systems*, 2017-Septe:3183–3189, 2017.

[RMSL11] Jorge Rios-Martinez, Anne Spalanzani, and Christian Laugier. Understanding human interaction for probabilistic autonomous navigation using risk-RRT approach. In *IEEE International Conference on Intelligent Robots and Systems*, 2011.

[RZBS09] Nathan Ratliff, Matt Zucker, J Andrew Bagnell, and Siddhartha Srinivasa. Chomp: Gradient optimization techniques for efficient motion planning. In *IEEE International Conference on Robotics and Automation*, pages 489–494, 2009.

[SC19] V. Sathiya and M. Chinnadurai. Evolutionary algorithms-based multi-objective optimal mobile robot trajectory planning. *Robotica*, 37(8):1363–1382, 2019.

[SGEM$^+$19] Mario Selvaggio, Amir Ghalamzan E., Rocco Moccia, Fanny Ficuciello, Bruno Siciliano, et al. Haptic-guided shared control for needle grasping optimization in minimally invasive robotic surgery. In *IEEE/RSJ International Conference Intelligent Robotic System*, 2019.

[SH15] C. Schindlbeck and S. Haddadin. Unified passivity-based cartesian force/impedance control for rigid and flexible joint robots via task-energy tanks. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 440–447, 2015.

[SH16] O. Salzman and D. Halperin. Asymptotically near-optimal rrt for fast, high-quality motion planning. *IEEE Transactions on Robotics*, 32(3):473–483, 2016.

[SLM$^+$19] Hang Su, Shuai Li, Jagadesh Manivannan, Luca Bascetta, Giancarlo Ferrigno, and Elena De Momi. Manipulability Optimization Control of a Serial Redundant Robot for Robot-assisted Minimally Invasive Surgery. *IEEE International Conference on Robotics and Automation*, 2019.

[Spo96] Mark W. Spong. Energy Based Control of a Class of Underactuated Mechanical Systems. *IFAC Proceedings Volumes*, 29(1):2828–2832, 1996.

[SR17] Hou Shunxiang and Ling Rui. Tracking control for underactuated VTOL aircraft. *Proceedings of the 29th Chinese Control and Decision Conference, CCDC 2017*, pages 1646–1650, 2017.

[Sra15] Suvrit Sra. Positive definite matrices and the S-divergence. *Proceedings of the American Mathematical Society*, 144(7):2787–2797, 2015.

[STMPG17] J. Sverdrup-Thygeson, S. Moe, K. Y. Pettersen, and J. T. Gravdahl. Kinematic singularity avoidance for robot manipulators using set-based manipulability tasks. In *1st Annual IEEE Conference on Control Technology and Applications, CCTA 2017*, 2017.

[SZS20] Jochen Stüber, Claudio Zito, and Rustam Stolkin. Let's push things forward: A survey on robot pushing. *Frontiers in Robotics and AI*, 7:8, 2020.

[TGT$^+$16] M. Talha, E. A. M. Ghalamzan, C. Takahashi, J. Kuo, W. Ingamells, and R. Stolkin. Towards robotic decommissioning of legacy nuclear plant: Results of human-factors experiments with tele-robotic manipulation, and a discussion of challenges and approaches for decommissioning. In *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 166–173, 2016.

[Tho99] M Thoma. *Variable structure systems, sliding mode and nonlinear control.* Springer, 1999.

[tPGSP17] Andreas ten Pas, Marcus Gualtieri, Kate Saenko, and Robert Platt. Grasp pose detection in point clouds. *The International Journal of Robotics Research*, 36(13-14), 2017.

[US03] Chris Urmson and Reid G. Simmons. Approaches for heuristically biasing rrt growth. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1178–1183. IEEE, 2003.

[VDAD10] Nikolaus Vahrenkamp, Martin Do, Tamim Asfour, and Rüdiger Dillmann. Integrated grasp and motion planning. In *IEEE International Conference on Robotics and Automation*, pages 2883–2888. IEEE, 2010.

[VKSK16] F. E. Viña B., Y. Karayiannidis, C. Smith, and D. Kragic. Adaptive control for pivoting with visual and tactile feedback. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 399–406, 2016.

[VZ11] Luc Vinet and Alexei Zhedanov. Rapidly-Exploring Random Trees: A New Tool for Path Planning. Technical Report 8, 2011.

[WGL$^+$19] Jiang Wei, Wu Gongping, Yu Lianqing, Li Hongjun, and Chen Wei. Manipulator multi-objective motion optimization control for high voltage power cable mobile operation robot. *Journal of Ambient Intelligence and Humanized Computing*, 37(10), 2019.

[WGY$^+$20] W. Wang, H. Gao, Q. Yi, K. Zheng, and T. Gu. An improved rrt* path planning algorithm for service robot. In *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, volume 1, pages 1824–1828, 2020.

[WM20] Jiankun Wang and Max Meng. Optimal path planning using generalized voronoi graph and multiple potential functions. *IEEE Transactions on Industrial Electronics*, PP:1–1, 01 2020.

[Woo20] James Woodfield. Cleaning up our nuclear past: faster, safer and sooner. `https://nda.blog.gov.uk/2020/01/10/how-much-radioactive-waste-is-there-in-the-uk/`, 2020.

[WTH17] Z. Wang, Y. Torigoe, and S. Hirai. A prestressed soft gripper: Design, modeling, fabrication, and tests for food handling. *IEEE Robotics and Automation Letters*, 2(4):1909–1916, 2017.

[Wv13] D. J. Webb and J. van den Berg. Kinodynamic rrt*: Asymptotically optimal motion planning for robots with linear dynamics. In *2013 IEEE International Conference on Robotics and Automation*, pages 5054–5061, 2013.

[WW99] J. T. Wen and L. S. Wilfinger. Kinematic manipulability of general constrained rigid multibody systems. *IEEE Transactions on Robotics and Automation*, 15(3):558–567, 1999.

[WWM$^+$19] Yancheng Wang, Xin Wu, Deqing Mei, Lingfeng Zhu, and Jianing Chen.

Flexible tactile sensor array for distributed tactile sensing and slip detection in robotic hand grasping. *Sensors and Actuators A: Physical*, 2019.

[XÖ08] Rong Xu and Ümit Özgüner. Sliding mode control of a class of underactuated systems. *Automatica*, 44(1):233–241, 2008.

[YL12] Zhu Yongxin and Fan Liping. On robust hybrid force/motion control strategies based on actuator dynamics for nonholonomic mobile manipulators. *Journal of Applied Mathematics*, (1), 2012.

[Yos85a] T. Yoshikawa. Dynamic manipulability of robot manipulators. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, pages 1033–1038, 1985.

[Yos85b] Tsuneo Yoshikawa. Manipulability of robotic mechanisms. *The International Journal of Robotics Research*, 4(2):3–9, 1985.

[YPW19] Yajue Yang, Jia Pan, and W. Wan. Survey of optimal motion planning. 2019.

[ZHL+17] Philipp Zech, Simon Haller, Safoura Rezapour Lakani, Barry Ridge, Emre Ugur, and Justus Piater. Computational models of affordance in robotics: a taxonomy and systematic classification. *Adaptive Behavior*, 25(5):235–271, 2017.

[ZRD+13] Matt Zucker, Nathan Ratliff, Anca D. Dragan, Mihail Pivtoraiko, Matthew Klingensmith, Christopher M. Dellin, J. Andrew Bagnell, and Siddhartha S. Srinivasa. CHOMP: Covariant Hamiltonian optimization for motion planning. *International Journal of Robotics Research*, 2013.

[ZSKW12] C. Zito, R. Stolkin, M. Kopicki, and J. L. Wyatt. Two-level rrt planning for robotic push manipulation. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 678–685, 2012.

[ZZL+18] Mingyue Zhang, Man Zhou, Hui Liu, Baiqiang Zhang, Yulian Zhang, and Hairong Chu. Friction compensation and observer-based adaptive sliding mode control of electromechanical actuator. *Advances in Mechanical Engineering*, 10(12):1–15, 2018.